**Homework 3: Basic SQL (Total Points: 100)**

Due: February 27, 11:59PM ET

**Problem 1: Database Keys (Points: 10)**

At the initial stage of designing a database for a fitness app we have identified the following data that we would like to store in a database. Please review the proposed data structure and answer the questions listed below.

Table name: TRAINING

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | INSERT\_TIMESTAMP | WORKOUT\_DATE | WORKOUT\_TYPE | WORKOUT\_DURATION | WORKOUT\_DISTANCE\_MILES | WORKOUT\_CITY | WORKOUT\_STATE | WORKOUT\_COMMENT | USER\_EMAIL | USER\_NAME | USER\_PAYMENT |
| 1 | 2021-02-04 10:33:12 | 1/12/2021 | Running | 1:05:34 | 8.12 | Cambridge | MA | Awesome run! | lisa@mymail.com | Lisa Simpson | VISA |
| 2 | 2021-02-05 15:30:02 | 1/3/2021 | Swimming | 0:22:10 | 0.5 | Boston | MA | Boring but necessary | wonka@yahoo.com | Willie Wonka | PAYPAL |
| 3 | 2021-02-06 20:26:52 | 2/2/2021 | Hiking | 3:12:09 | 6.41 | Burlington | VT |  | jhn@work.org | John Doe |  |
| 4 | 2021-02-08 12:23:42 | 2/2/2021 | Walking | 3:15:33 | 6.48 |  |  | Scenic views | jdoe@vz.com | Jane Doe | MC |
| 5 | 2021-02-09 6:20:32 | 2/3/2021 | Weight Lifting | 0:45:01 |  | New Haven | CT | Nailed it! | jdoe@vz.com | Jane Doe | MC |
| 6 | 2021-02-10 11:17:03 | 10/13/2021 | Running | 1:58:12 | 26.2 | Chicago | IL | Set my PR on this one! | jr@yahoo.com | Jack Rabbit |  |

**Question 1.1:** Which field(s) could be selected as Candidate Keys and why?

Field “ID” could be selected as Candidate key, for ID->INSERT\_TIMESTAMP, WORKOUT\_DATE, WORKOUT\_TYPE, WORKOUT\_DURATION, WORKOUT\_DISTANCE\_MILES, WORKOUT\_CITY, WORKOUT\_STATE, WORKOUT\_COMMENT, USER\_EMAIL, USER\_NAME, USER\_PAYMENT

Fileds “USER\_NAME”, “WORKOUT\_DATE”, “WORKOUT\_TYPE”, “INSERT\_TIMESTAMP” could be selected as candidate key. For USER\_EMAIL,WORKOUT\_DATE, WORKOUT\_TYPE, NSERT\_TIMESTAMP -> ID, WORKOUT\_DURATION, WORKOUT\_DISTANCE\_MILES, WORKOUT\_CITY, WORKOUT\_STATE, WORKOUT\_COMMENT, USER\_NAME, USER\_PAYMENT

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Question 1.2:** Among the keys you listed in your answer above, which would be considered Composite Keys?

\_ USER\_EMAIL,WORKOUT\_DATE, WORKOUT\_TYPE, NSERT\_TIMESTAMP would be consider as composite key. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Question 1.3:** Which one of the candidate keys would you select to be the Primary Key and why?

I will select “ID” to be the primary key of the training table. Because it identifies user’s every workout record.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Question 1.4:** List all of the fields which could be stored in a separate table? Why would you store them in a separate table?

I think the fields could be stored in User table: USER\_EMAIL, USER\_NAME, USER\_PAYMENT.

And the fields could be stored in Workout table: ID, INSERT\_TIMESTAMP, WORKOUT\_DATE, WORKOUT\_TYPE, WORKOUT\_DURATION, WORKOUT\_DISTANCE\_MILES, WORKOUT\_CITY, WORKOUT\_STATE, WORKOUT\_COMMENT, USER\_EMAIL

Because every user has a unique email. And every user can have many workouts. So separate user information from training, It can reduce the redundancy of user information

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Question 1.5:** Suppose you were told that instead of storing the user's city and state in the same table, you need to reference the location stored in a separate **locations** table (example shown below). Describe how you would reference the **locations** table from your **training** table? Which fields would you drop and/or add from/to your workout table? What would be your foreign key?

|  |  |  |  |
| --- | --- | --- | --- |
| LOCATION\_ID | CITY | STATE | ZIP\_CODE |
| 34561 | Boston | MA | 02101 |
| 30999 | New York | NY | 10001 |

\_I would add the field “LOCATION\_ID”\_to training table and reference the locations table use attribute “LOCATION\_ID”. Then I would drop fields “WORKOUT\_CITY “ and “WORKOUT\_STATE” from training table. The field “LOCATION\_ID” will be the foreign key of training table.\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Problem 2: Database Types (Points: 10)**

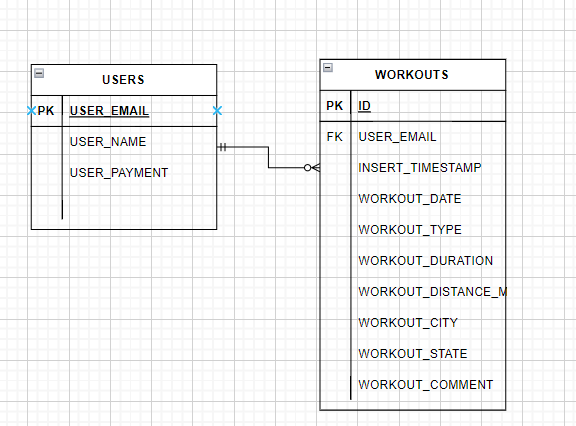
In the table below, please specify which data type you would choose for each field (e.g. DATE, TIME, INT, VARCHAR(10), etc…):

|  |  |
| --- | --- |
| **Field Name** | **Field Type** |
| ID | INT |
| INSERT\_TIMESTAMP | TIMESTAMP |
| WORKOUT\_DATE | DATE |
| WORKOUT\_TYPE | VARCHAR(30) |
| WORKOUT\_DURATION | TIME |
| WORKOUT\_DISTANCE\_MILES | FlOAT |
| WORKOUT\_DISTANCE\_MILES | FlOAT |
| WORKOUT\_CITY | VARCHAR(30) |
| WORKOUT\_STATE | CHAR(2) |
| WORKOUT\_COMMENT | TEXT |
| USER\_EMAIL | VARCHAR(30) |
| USER\_NAME | VARCHAR(30) |

**Problem 3: Relational Database Design (Points: 20)**

Redesign the database by breaking up the TRAINING table into two tables (USERS and WORKOUTS) in such a way that the WORKOUTS table would reference the USERS table using a foregin key. The payment information is the user's way to pay the monthly subscription fees for the use of the application. For simplicity we assume that each user can have only one payment method on file, and it can be described by a single word such as "VISA", "MASTERCARD", etc. Some users might not have payment information on file.

Draw the diagram of your new design below and provide a short verbal explanation of your design. Are there any constraints that you would impose on the values of any fields, and why?:



For every user has a unique email. And every user can have many workouts. And a workout record must belong to a user.

USERS(USER\_EMAIL,USER\_NAME, USER\_PAYMENT)

WORKOUTS (ID, USER\_EMAIL, INSERT\_TIMESTAMP, WORKOUT\_DATE, WORKOUT\_TYPE, WORKOUT\_DURATION, WORKOUT\_DISTANCE\_MILES, WORKOUT\_CITY, WORKOUT\_STATE, WORKOUT\_COMMENT)

There is a foreign key constraint, the field USER\_EMAIL of WORKOUTS references USERS(USER\_EMAIL).

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Problem 4: Create Your Database (Points: 20)**

1. Using valid SQL:

* Write a statement to create a new database called "training" according to the design you created in Problem 3.
* Write statements to create your two tables

Make sure the relationships between the two tables are properly defined with a foreign key.

Provide all SQL statements used:

create database training;

use training;

create table USERS(

USER\_EMAIL varchar(30) primary key,

USER\_NAME varchar(30),

USER\_PAYMENT varchar(30)

);

create table WORKOUTS(

ID int,

USER\_EMAIL varchar(30),

INSERT\_TIMESTAMP timestamp,

WORKOUT\_DATE date,

WORKOUT\_TYPE varchar(30),

WORKOUT\_DURATION time,

WORKOUT\_DISTANCE\_MILES float,

WORKOUT\_CITY varchar(30),

WORKOUT\_STATE char(2),

WORKOUT\_COMMENT text,

foreign key (USER\_EMAIL) references USERS(USER\_EMAIL)

);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Write a SQL statement or statements to Insert all of the sample data from the table used in Problem 1 into your tables. Make the necessary adjustments required to set up relationship between your two tables.

Paste one of the INSERT statements into USERS table:

insert into USERS values ('lisa@mymail.com', 'Lisa Simpson', 'VISA');

insert into USERS values ('wonka@yahoo.com', 'Willie Wonka', 'PAYPAL');

insert into USERS values ('jhn@work.org', 'John Doe', NULL);

insert into USERS values ('jdoe@vz.com', 'Jane Doe', 'MC');

insert into USERS values ('jr@yahoo.com', 'Jack Rabbit', NULL);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
Paste one of the INSERT statements into WORKOUTS table:

insert into WORKOUTS values (1,'lisa@mymail.com', '2021-02-04 10:33:12','2021-01-12', 'Running', '1:05:34', 8.12, 'Cambridge','MA','Awesome run!');

insert into WORKOUTS values (2,'wonka@yahoo.com', '2021-02-05 15:30:02','2021-01-03','Swimming','0:22:10', 0.5,'Boston','MA', 'Boring but necessary');

insert into WORKOUTS values (3,'jhn@work.org', '2021-02-06 20:26:52','2021-02-02','Hiking','3:12:09',6.41,'Burlington','VT',null);

insert into WORKOUTS values (4,'jdoe@vz.com', '2021-02-08 12:23:42','2021-02-02','Walking', '3:15:33', 6.48,null, null, 'Scenic views');

insert into WORKOUTS values (5,'jdoe@vz.com', '2021-02-09 6:20:32', '2021-02-03','Weight Lifting','0:45:01',null,'New Haven','CT','Nailed it!');

insert into WORKOUTS values (6,'jr@yahoo.com', '2021-02-10 11:17:03', '2021-10-13','Running','1:58:12',26.2, 'Chicago','IL','Set my PR on this one!');  
   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

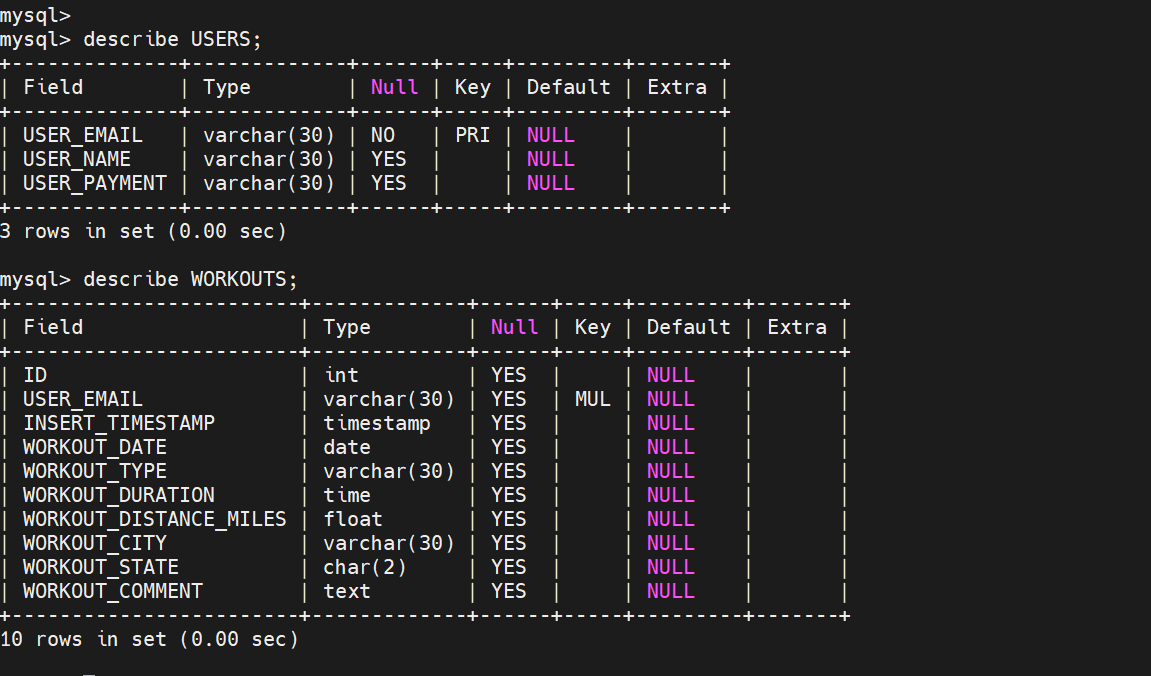
**Problem 5: SQL Queries (Points: 40):**

Paste the SQL statement **and the screenshot of the results** for each of the following

1. DESCRIBE for each of the 2 tables:

describe USERS;

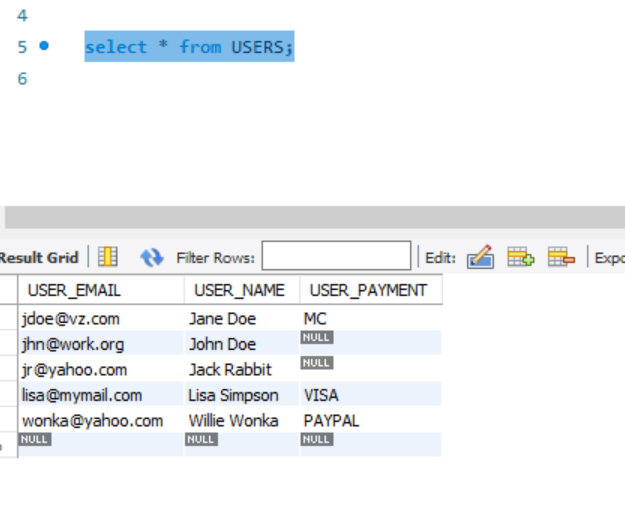
describe WORKOUTS;



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. SELECT \* from each of the tables (two separate selects, one for each table):

select \* from USERS;



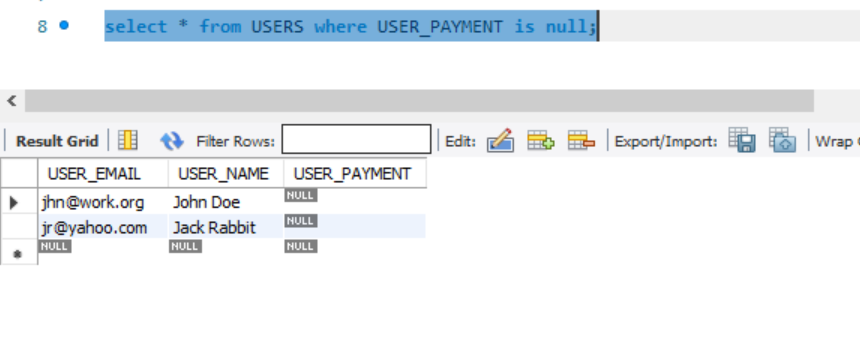
select \* from WORKOUTS;



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. From table USERS, list all users who do not have their payment information on file.

select \* from USERS where USER\_PAYMENT is null;



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

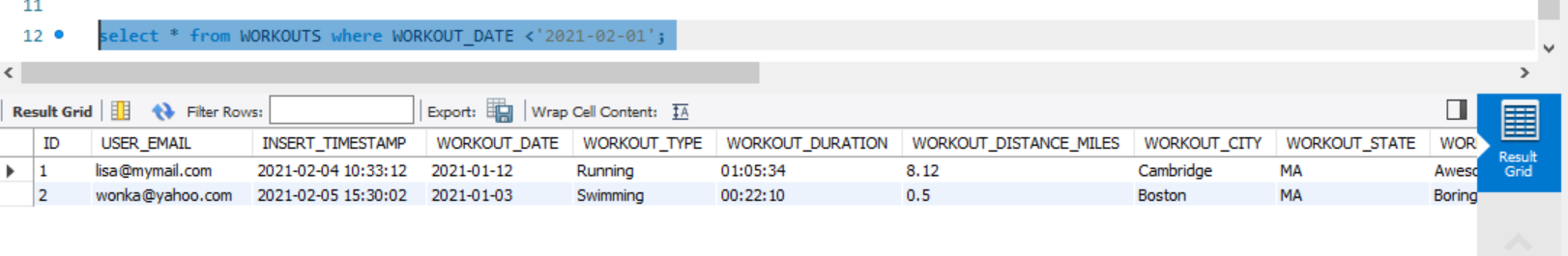
1. Update the users with no payment information to set the the value to "FREE"

update USERS set USER\_PAYMENT='FREE' where USER\_PAYMENT is null;  


\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

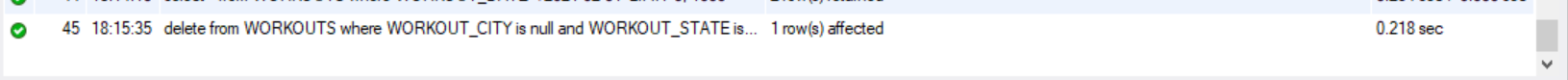
1. From table WORKOUTS list all the workouts that occurred before February 1, 2021.

select \* from WORKOUTS where WORKOUT\_DATE <'2021-02-01';

  
  
 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Delete the workouts that do not have the location information.

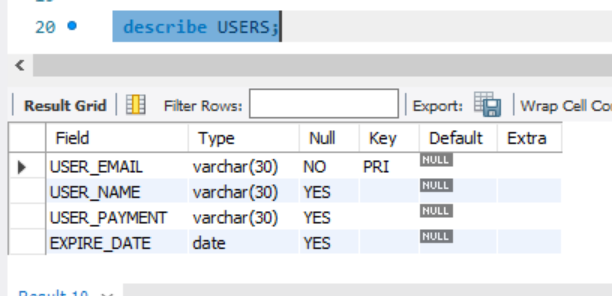
delete from WORKOUTS where WORKOUT\_CITY is null and WORKOUT\_STATE is null;

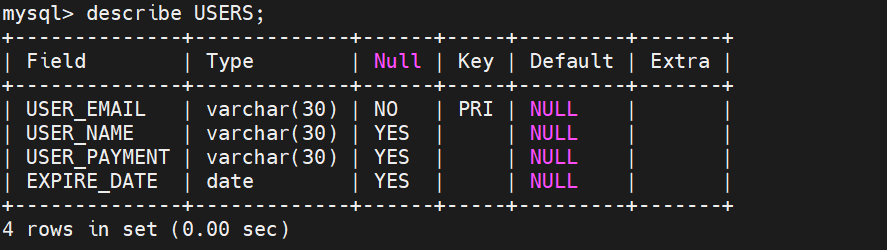


\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. ALTER the table USERS to store the date when the payment information expires. Paste the ALTER statement and the results of the DESCRIBE statement executed for that table after your changes:

alter table USERS add column EXPIRE\_DATE date;

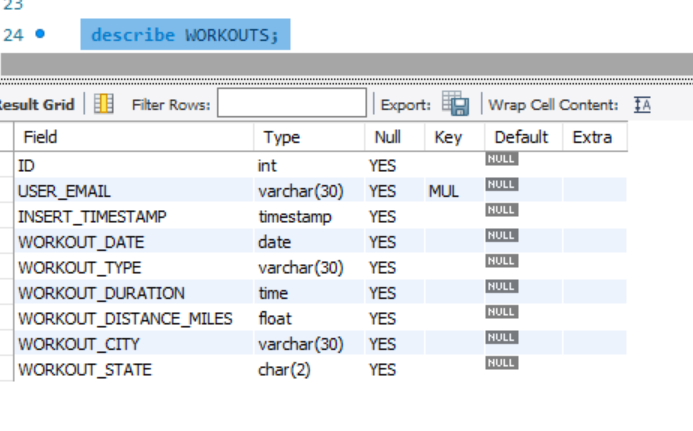


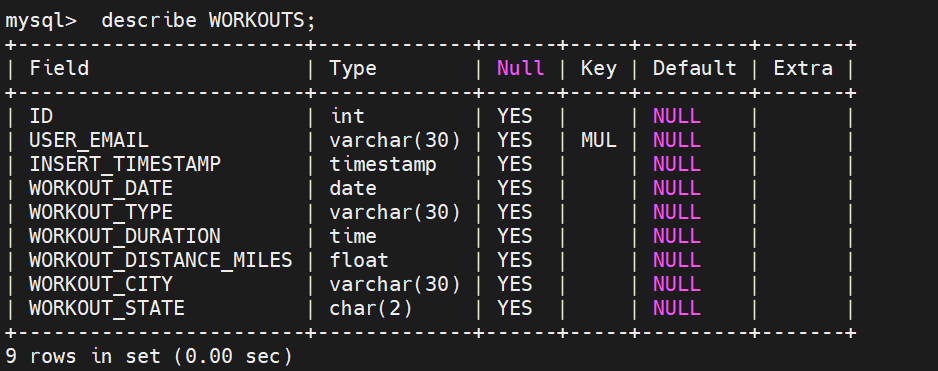


\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Remove the field WORKOUT\_COMMENT from the WORKOUTS table. Paste the ALTER statement that does that and the screenshot of the DESCRIBE statement executed for that table after you removed the field:

alter table WORKOUTS drop column WORKOUT\_COMMENT;





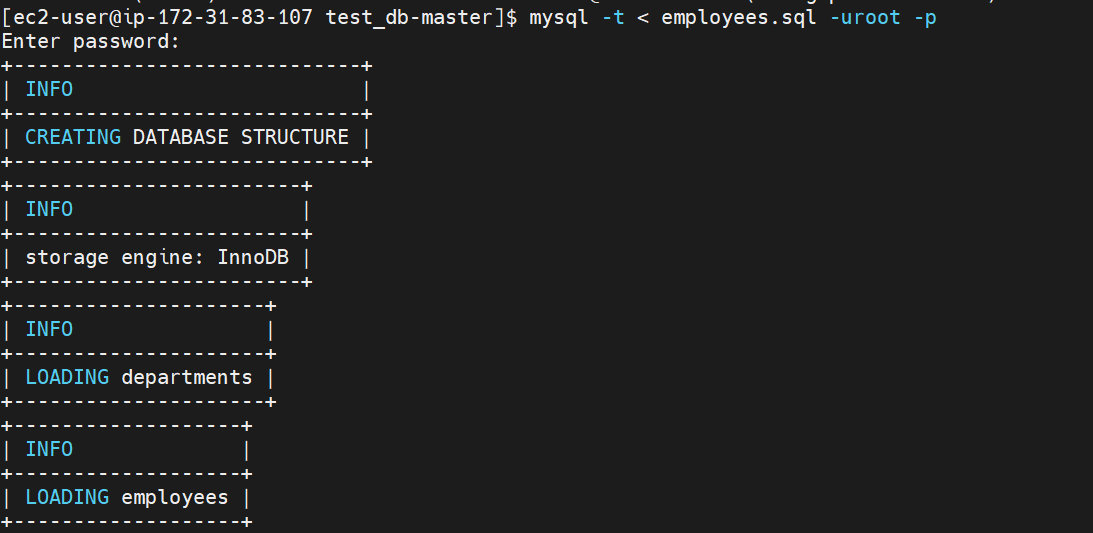
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Bonus Problem: Download a sample database from the MySQL website and import it into your database instance running on the EC2 instance. Write and execute a few queries as described below. (Points: 10)**

1. Read the description of the employees database on the MySQL website:

<https://dev.mysql.com/doc/employee/en/>

1. Connect to your EC2 instance where you have the MySQL DB installed and download the zipped database to your computer from GitHub (you don’t need a GitHub account to do that): <https://github.com/datacharmer/test_db> - use the wget command (typed on a single line):  
     
   wget --output-document test\_db-master.zip https://github.com/datacharmer/test\_db/archive/master.zip
2. On the EC2 instance, follow the Install instructions to import your DB into your MySQL instance:  
   <https://dev.mysql.com/doc/employee/en/employees-installation.html>   
   *Hint 1: In the second step of that instruction, the correct option is   
   set storage\_engine = InnoDB;  
     
   Hint 2: modify the import command by adding "-uroot -p", like this - enter your MySql root's password when asked  
   mysql -uroot -p -t < employees.sql*



1. Validate that your installation succeeded running the following command:  
     
   time mysql -uroot -p -t < test\_employees\_sha.sql  
     
   Paste the screenshot of the output of the test below:

|  |
| --- |
|  |

1. Write and execute the following queries (and paste your queries and screenshots with results):  
   *Note: use the diagram at* [*https://dev.mysql.com/doc/employee/en/sakila-structure.html*](https://dev.mysql.com/doc/employee/en/sakila-structure.html) *to understand the structure of the data. Remember that you have to login to MySql as root user now.*

6.1. Using the **employees** database and based on the data in the **employees** table figure out what is the percentage of male vs. female employees working in the company.

Male Employees: % Female Employees: %

select (

select count(\*) as male\_count

from employees

where gender='M'

)/(select count(\*) from employees) as male\_percent,

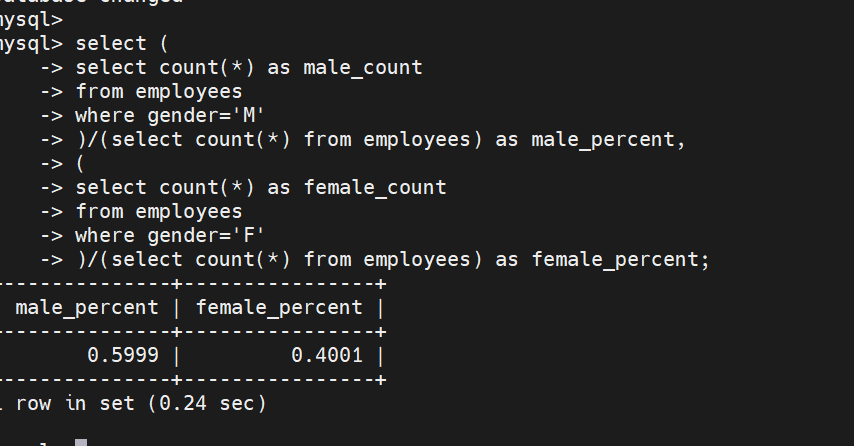
(

select count(\*) as female\_count

from employees

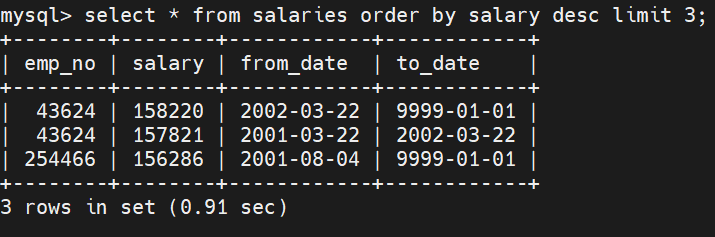
where gender='F'

)/(select count(\*) from employees) as female\_percent;  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



6.2. What are the 3 highest salaries in the organization:

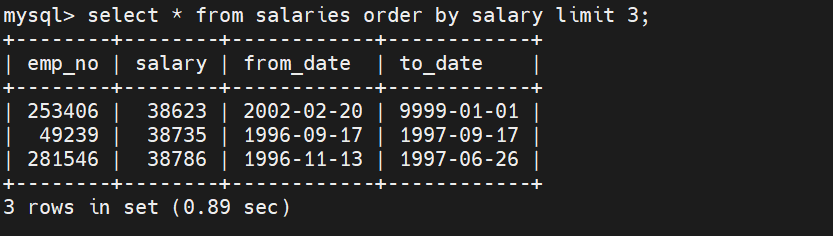
select \* from salaries order by salary desc limit 3;



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6.3. What are the 3 lowest salaries in the organization:

select \* from salaries order by salary limit 3;



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Note: You may encounter an Out Of Memory error trying to run SQLs for 6.2 and 6.3, but using "LIMIT" will help to avoid it.