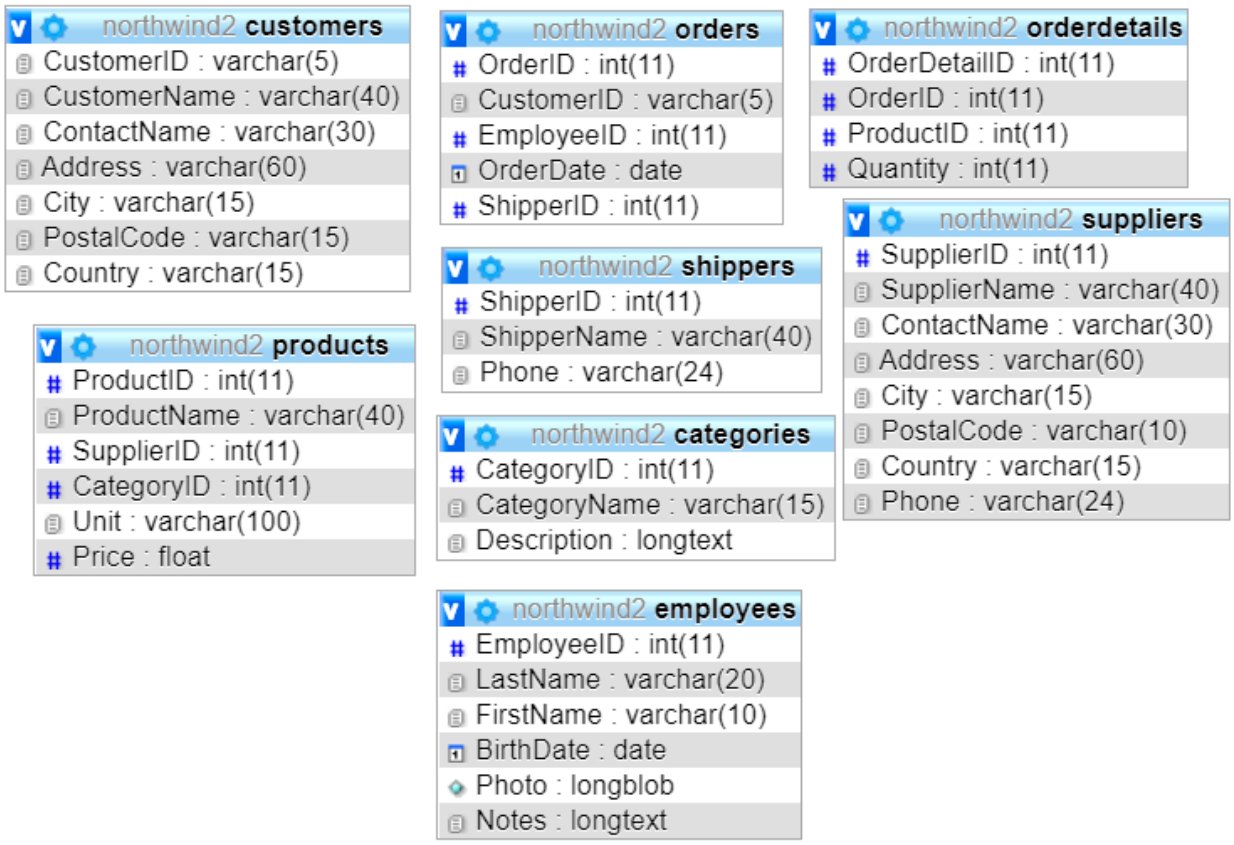
100 RDMB Database questions

With answer key and detailed explanation

Introduction

Consider the NorthWind Database below when answering the questions provided.



1 Assuming CustomerID being the primary key of Customers table, list the CustomerName of the most recently entered record

Do you have to make any assumption?

Answer:

Since there is no attribute indicating the time of creation of that record, there is no way to tell what record was the last entered record.

Note: We cannot assume the greatest value of CustomerID corresponds to the last (most recently) entered record.

Counter example:

delete FROM [Customers] where CustomerID = 1
Followed by

insert into Customers values (1,"Jessica ", "My contact", "My address", "My city", "V68 2K3", "Canada")

2 Can we list the CustomerName of the most recently entered record In the customers table, assuming CustomerID being Auto increment ?

Answer:

The largest CustomerID would represent the most recently entered record only if we assume there was no record deleted
Not deleting is the necessary condition, not the enough condition:
Two counter example :
Example 1:

What if you delete the last entered record and then look for the customer with the highest id; Is that customer the most recently entered one?
Now, is the largest customerID representing the last entered record?

Example 2:
run these two queries

delete from customers where customerID = 1
insert into [Customers] values(1,"John", "John Lo", "Hotel tower", "Burnaby", "1g2t3", "Canada")

Now is the largest customerID representing the last entered record? No! the last entered record is actually a customer with the smallest value! CustomerID=1

3 Using the Customers table, list the number of customers from each city

Answer:

SELECT count(CustomerID) as nOfCustomers , city FROM [Customers] group by city order by nOfCustomers desc
Note: pay attention to the next follow up question

4 What exceptions do you think were needed to be considered when answering question 3?

Answer:

Your SQL statement of question 3 has to cover exceptions such as

Exception 1 - What if there were same name cities but in different countries?
To verify, try your query after this insertion
insert into Customers values (200 ,"Jessica ", "My contact", "My address", "London", "V68 2K3", "Canada")
Solution to Exception 1: replacing "group by city" by " group by city, country" solves the problem
SELECT city, count(customerid) FROM [Customers] group by city, country

Exception 2: - what if the same city were spelled differently each time entered?
Solution to Exception 2: a better solution is have a separate table for the city table, separate one for the country ...

5 Does the table customers comply with 1NF?

Answer:yes, all attributes are single values

6 Does the table customers comply with 2NF rules?

Answer: Remember, a relation is in second normal form if and only if it is in 1NF and every non key attribute is fully functionally dependent on the entire primary key.
So if I grab a customerID, can I get to a country ? Does customerID yield a country? no. For example the attribute City does not functionally depend on the customerID
Therefore the column (attribute) city does not funionaly depend on the primary key customerID and thus this table is not in 2NF

7 To account for the exception 4.2 what better solution would you suggest? (a solid solution to make sure all instances of same city spelled and typed same way)

Answer: The suggestion is to make the customer table comply with 2NF.

That means instead of entering each City name each time entering a record, create a separate table named Cities and enter each tity in that table only once. This way instead of entering City each time for each new record of the customer, you just enter CityID. Same thing for country etc

8 Will the SQL statement below execute for this table?

```
CREATE TABLE Persons (
    Personid int,
    LastName varchar(255) NOT NULL,
    PRIMARY KEY (Personid)
);
```

INSERT INTO `persons` VALUES (12, "McDonald")

Answer: Yes, provided there is already no record with Personid of 12

8.1 what if there were some records entered already ?

Answer: only if there is no record already with the id 12 (the engine does not let us violate the uniqueness constraint of the primary key)

9 Considering the table in question 8, will this query work?

INSERT INTO `persons` VALUES ("McDonald 2")
Answer: no! We have to specify both values so the engine knows which value is for which column

10 Following question 8, will the query below execute ?

INSERT INTO persons(Personid,LastName) VALUES (NULL, "McDonald 2")
Answer: yes. In our table creation, we did not expelisity say the primary key cannot be null.

Question: Can we enter another record with the primary key to be null for this table?
INSERT INTO persons(Personid,LastName) VALUES (NULL, "McDonald 3")

Not in mysql, but interestingly in SLQlite which is used for w3school demo db, having multiple records with primary key null was possible!

11 Following question 8, will this query work?

Query: INSERT INTO persons(LastName) VALUES ("McDonald 2")

Answer: No! The primary key is not auto increment

12 create a table with auto increment id and "not null". Now, do you have to specify the value of the primary key when you insert data into it?

Answer:

```
CREATE TABLE Persons (
    Personid int NOT NULL AUTO_INCREMENT,
    LastName varchar(255) NOT NULL,
    PRIMARY KEY (Personid)
);
```

Answer:

No since it is a primary key .since it will automatically generate a unique value for the primary key each time a new record is added

13 considering the table below, will this query work?

Query: Insert into Persons values ("Brown")

```
CREATE TABLE Persons (
    Personid int NOT NULL AUTO_INCREMENT,
    LastName varchar(255) NOT NULL,
    PRIMARY KEY (Personid)
);
```

Answer: No! The engine is confused and does not know we intend to give the value "Brown" to what attribute

14 considering the table below, will this query work?

```
CREATE TABLE Persons (
    Personid int NOT NULL AUTO_INCREMENT,
    LastName varchar(255) NOT NULL,
    PRIMARY KEY (Personid)
);
```

Query: Insert into Persons(`LastName`) values ("Brown")

Answer: Yes. The reason is that the primary key is auto increment.

15 Considering the table below, add a new attribute password with enforcing the password to be at least 8 characters long.

```
ALTER TABLE Persons

ADD password varchar(255) DEFAULT NULL;
```

Answer:
/* MySQL Solution */
ALTER TABLE Persons
ADD CHECK (LENGTH(password) >= 6);

Testing in phpMyAdmin
INSERT INTO persons VALUES(100, "Green", "12345")
Error

SQL query:

INSERT INTO persons VALUES(100, "Green", "12345")

MySQL said:

#4025 - CONSTRAINT `CONSTRAINT_1` failed for `TestDB`.`persons`

16 Can you come up with an alternative solution to force passwords to be more than 6 characters long, without using length function ?

Answer: Try using wildcard

```
ALTER TABLE persons

ADD CHECK (`password` LIKE ('_______%'));
```

17- add a new column BirthDate with type date and no default value to the table below.

What will be the value of BirthDate for the records already entered?

```
CREATE TABLE Persons (
    Personid int NOT NULL AUTO_INCREMENT,
    LastName varchar(255) NOT NULL,
    PRIMARY KEY (Personid)
);
```

Answer:in mySQL:

```
ALTER TABLE Persons

ADD BirthDate date;
```

The value of BirthDate column for existing records will be '0000-00-00' according to our observation on phpMyAdmin. Update: it will be NULL in newer versions (as of 5.2.0)

18- Assume we want to add a check constraint to the table below to make sure only records of users whose age is more than 18 can be entered into the DB. Can we add a new constraint after the table is already created ?

Answer: yes ALTER TABLE tableName ADD CHECK (condition);

19. If the answer to question 18 is a yes, what will happen to the existing records whose BirthDate does not pass the CHECK?

Answer: try it in PhpMyAdmin or workbench

20 - create a one-one relationship between a person and its birth certificate . How would you enforced the 1:1 relationship by adding the required constraints

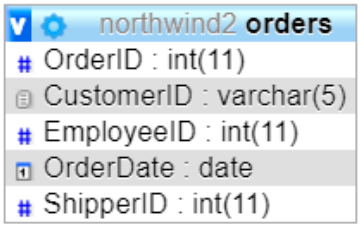
Answer:

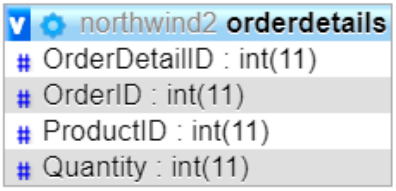
We make the foreign key unique

```
CREATE TABLE Person(

PersonID  int NOT NULL AUTO_INCREMENT,

Name  varchar(255) NOT NULL,

PRIMARY KEY (PersonID)

);

CREATE TABLE BirthCertificate(

BirthCertificateID  int NOT NULL AUTO_INCREMENT,

PersonID  int NOT NULL,

IssueDate  date NOT NULL,

PRIMARY KEY (BirthCertificateID),

FOREIGN KEY (PersonID) REFERENCES Person(PersonID),

        UNIQUE KEY unique_patient (PersonID)

);
```

We will make the foreign key unique

21 Write a query that finds, for each customer X, another customer Y who has also ordered at least on the same date in common with X. Find all such pairs of Customers (X, Y) and against each pair, the number of overlapping dates.





The query should thus have three columns. Order the results by the number of overlapping dates

your result set must have three columns

CustomerXName, CustomerYName, nOfOverlappingDates

Answer:

As a tiny step to verify your answer, I want to bring to your attention the number of orders happened in the same date by running this query:

SELECT OrderDate, count (OrderID) as overlappingOrders FROM [Orders] group by OrderDate order by overlappingOrders desc

Now, lets list the customer IDs (not the customer names) who happen to place orders on the same day

Note: Here the assumption was any two customers will only have a common date with their others in just one day .

```
SELECT a.CustomerID as 'CustomerXid', B.CustomerID as 'CustomerYid', a.OrderDate
FROM Orders as a, Orders as b
WHERE a.OrderDate = b.OrderDate AND a.CustomerID<b.CustomerID order by CustomerXid
```

As you see, customers 4 and 5 have only one common order date. Let's make this two by inserting two records for customers 4, 5 for another common date:

insert into orders(OrderID, CustomerID, EmployeeID, OrderDate, ShipperID) values (11000, 4,9,'2020-02-02', 1)

insert into orders(OrderID, CustomerID, EmployeeID, OrderDate, ShipperID) values (11001, 5,9,'2020-02-02', 1)

Lets run the query above again:

So our final query has to list two common order dates for the customers with ID 4 and 5 now. This way we can verify our query better.

Let's do it in two steps:

Step 1: List the customers IDs of customers who ordered in same dates,

Step 2: find out the name of those customers from step 1

Step 1:

```
SELECT o1.CustomerID as CustomerID1,
       o2.CustomerID as CustomerID2,
       COUNT( o1.OrderDate) as NumberOfCommonDates
 FROM Orders o1 JOIN
     Orders o2
     ON o1.OrderDate = o2.OrderDate AND
        o1.CustomerID < o2.CustomerID
 GROUP BY o1.CustomerID, o2.CustomerID
 ORDER BY NumberOfCommonDates DESC
```

Step 2:
Now we want to join the resulting table to the customers table to find out the names of the customers whose ID was in the resulting table of Step 1. Consider step 1 (highlighted) a subquery to be used in step 2 like this:

```
select c1.CustomerName,c2.CustomerName,CustomerID1, CustomerID2,
NumberOfCommonDates FROM customers as c1,customers as c2,
(SELECT o1.CustomerID as CustomerID1,
       o2.CustomerID as CustomerID2,
       COUNT( o1.OrderDate) as NumberOfCommonDates
 FROM Orders o1 JOIN
```

```
    Orders o2
    ON o1.OrderDate = o2.OrderDate AND
       o1.CustomerID < o2.CustomerID
 GROUP BY o1.CustomerID, o2.CustomerID
 ORDER BY NumberOfCommonDates DESC)
WHERE C1.CustomerID = CustomerID1 AND C2.CustomerID = CustomerID2
```

Question: what would be an exception case? Let's make sure it also works for exceptions! Such as when two customers put orders twice in the same day . Let's then insert these:

insert into orders(OrderID, CustomerID, EmployeeID, OrderDate, ShipperID) values (11002, 4,9,'2020-02-02', 1)

insert into orders(OrderID, CustomerID, EmployeeID, OrderDate, ShipperID) values (11003, 5,9,'2020-02-02', 1)

The answer is wrong!! Since we repeated the same date twice, we did not want to see the NumberOfCommonDates to be any different this time! (no new date was added). Therefore we have to modify our answer: Guess how?

Answer: by adding distinct

```
SELECT o1.CustomerID as CustomerID1,
      o2.CustomerID as CustomerID2,
      COUNT(DISTINCT o1.OrderDate) as NumberOfCommonDates
FROM Orders o1 JOIN
    Orders o2
    ON o1.OrderDate = o2.OrderDate AND
      o1.CustomerID < o2.CustomerID
GROUP BY o1.CustomerID, o2.CustomerID
ORDER BY NumberOfCommonDates DESC;
```

Therefore the final answer becomes:

```
select c1.CustomerName as CustomerName1, c2.CustomerName as
CustomerName2,CustomerID1, CustomerID2, NumberOfCommonDates
FROM customers as c1,customers as c2,
      (SELECT o1.CustomerID as CustomerID1,
            o2.CustomerID as CustomerID2,
            COUNT( DISTINCT o1.OrderDate) as NumberOfCommonDates
        FROM Orders o1 JOIN
            Orders o2
            ON o1.OrderDate = o2.OrderDate AND
                o1.CustomerID < o2.CustomerID
        GROUP BY o1.CustomerID, o2.CustomerID
        ORDER BY NumberOfCommonDates DESC)
WHERE
      C1.CustomerID = CustomerID1 AND C2.CustomerID = CustomerID2
```

Q: Do we have to use join to write this query?

Answer: no, you could write it without using join statement. Try it for yourself

22 Which query might run slower?

SELECT **CustomerName** FROM Customers, Orders, OrderDetails

vs

SELECT **CustomerName** FROM Customers

Answer: the former. It joins all the tables first (say each table size is 100 records, it creates a resulting table of size 100x100X100 = 1,000,000 records which contains the union of columns of all tables)Then grabs the column CustomerName

23 Why in question 22 we explicitly said "might" run slower?

Answer: other tables, Orders, OrderDetails might be empty

24 Assuming the Customers table has 7 columns, the Orders table only 5 columns, predict how many columns this SQL statement would generate ?

SELECT * FROM Customers, Orders, OrderDetails

Answer: When using the resulting table contains the **union** of columns of all tables involved . that means equal or less than 5+7 (some columns such as customerID is common column name)

25 Which query might run slower?

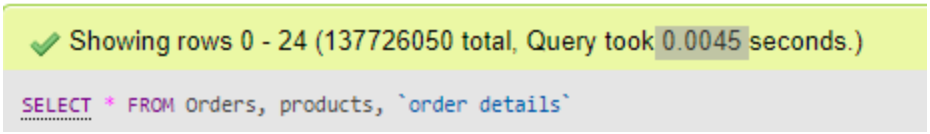
SELECT * FROM Customers, Orders, OrderDetails

vs

SELECT **CustomerName** FROM Customers, Orders, OrderDetails

Answer: both tables take almost the same time to run. For the latter we need less memory to store the result though

Note: in phpMyAdmin you can see how much the execution of your SQL query takes:



26 What result these SQL statements generate?

SELECT count(**CustomerName)** FROM Customers, Orders

SELECT count(*****) FROM Customers, Orders

27 how can you copy one (or more) column(s) from another table to another table?

Answer: in lecture notes, also:

```
INSERT INTO Customers (CustomerName, ContactName, Address, City,
PostalCode, Country)
SELECT SupplierName, ContactName, Address, City, PostalCode, Country FROM
Suppliers;
```

28 Is the table `customers` in 1NF? Is it in 2NF?

Yes, it is 1NF
Not 2NF

29 Is this table below in 2NF?

ScoreID	StdID	SubjectID	Mark	InstructorName
1	7001	1	87	Dr. Smitch
2	7001	2	56	Dr. Green
3	7441	2	76	Dr. Green
4	9001	11	87	Dr. Singh

Answer:
It is not 2NF

30 Provide an example of composite key in an M:M table

Answer:

```
CREATE TABLE `orderDetails` (
  `OrderDetailID` int(11) NOT NULL,
  `OrderID` int(11) NOT NULL,
  `ProductID` int(11) NOT NULL,
  `Quantity` int(11) NOT NULL DEFAULT 0,
  PRIMARY KEY (`OrderID`,`ProductID`)
)
```

The composite key is `PRIMARY KEY(`OrderID`,`ProductID`).`

31 Provide an example of union and left join of two tables with null values

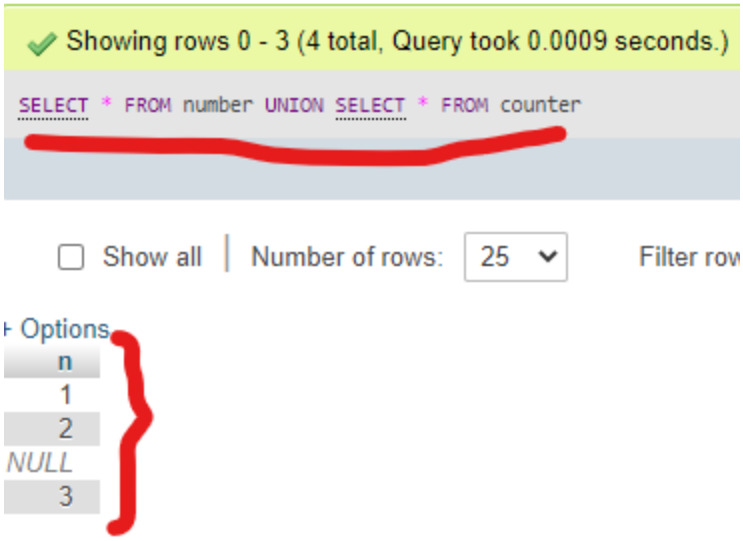
Answer:

```
CREATE TABLE number (
    n int
);
```

```
CREATE TABLE counter(
    n int
);

insert into number values (1),(2),(null),(null);
insert into counter values (2),(null), (3)
```

Now try inner join, left join, union etc and observe the resulting table.
For example:

```
SELECT * FROM number UNION SELECT * FROM counter;
```



32 consider the table below.

```
CREATE TABLE `customers` (
  `CustomerID` varchar(5) NOT NULL,
  `CustomerName` varchar(40) NOT NULL,
  `ContactName` varchar(30) DEFAULT NULL,
  `Address` varchar(60) DEFAULT NULL,
  `City` varchar(15) DEFAULT NULL,
   PostalCode varchar(15) DEFAULT NULL,
  `Country` varchar(15) DEFAULT NULL,
  PRIMARY KEY (`CustomerID`)
```

```
)
```

Give two counterexamples where the record with highest customerID does not represent the last entered record

Answer: Two counter examples :
Example 1:

What if you delete the last entered record and then look for the customer with highest id. is that customer the most recently entered record?
Now is the largest customerID representing the last entered record?

Example 2:
run these two queries

delete from customers where customerID = 1
insert into [Customers] values(1,"John", "John Lo", "Hotel tower", "Burnaby", "1g2t3", "Canada")

Now is the largest customerID representing the last entered record? No! the last entered record is actually a customer with the smallest CustomerID value! CustomerID=1 is actually the last entered record

33 Using SQL statements how would you create a database of two tables, citizen and passport?

```
CREATE TABLE Passport(
    PassportNumber int NOT NULL AUTO_INCREMENT,
    DateIssued date,
    PRIMARY KEY (PassportNumber)
);
CREATE TABLE Person (
    PersonId int NOT NULL AUTO_INCREMENT,
    Name varchar(255) NOT NULL,
    PassportNumber int,
```

```
    PRIMARY KEY (Personid),
    FOREIGN KEY (PassportNumber) REFERENCES Passport(PassportNumber)
);
```

34 Can a foreign key be null ?

Can one of the FKs in a table be null?

Answer:
Yes, foreign keys can be null in a table. For example in the tables below

There might be a person with no passport. Then the passportNumber of that person will be null

35 Write a query to list the second highest price from the table products

Answer:

```
SELECT max( price) FROM products where price not in (SELECT max(
price) FROM Products )
```

36 Write a query to select the name of the product(s) with the second highest price

Answer:

```
SELECT * FROM Products where price in (SELECT max( price) FROM
products where price not in (SELECT max( price) FROM Products ))
```

37 Assuming there is a 7th column named: "Salary" in the "Employees" table

Write a SQL query to fetch the list of employees with the same salary.

Answer:
SELECT a.FirstName as fname1, a.LastName as lname1, b.FirstName as fname2, b.LastName as lname2 FROM Employees a, Employees b

WHERE a.EmployeeID < b.EmployeeID

AND a.salary = b.salary

38 without using trigger, add a constraint for the table below so that doctors younger than 20 years old cannot be entered:

```
CREATE TABLE `doctor` (
  `dateOfBirth` datetime NOT NULL
);
```

Answer:
DATE CHECK (DATEDIFF(CURRENT_DATE(), dateofBirth) >= 8760)
test :
insert into doctor values ('1837-06-20')

Will this be entered?

insert into doctor values ('2000')

insert into doctor values ('2010')

dateOfBirth
1837-06-20 00:00:00
0000-00-00 00:00:00
0000-00-00 00:00:00
0000-00-00 00:00:00
0000-00-00 00:00:00

39 write a constraint to prevent entering record where the number of entered record to be less than 100

I.e. not more than 100 records can be entered in this table

40 Suppose there is a table named Players in our DB with multiple columns including PlayerID (PRIMARY KEY), Country (of type CHAR(20)), City (of type CHAR(20)) and ZipCode (of type CHAR(6)) and weight (of type SMALLINT(3))

Write a Query to select all records from the Players table and to sort the result alphabetically, first by the column Country, then, by the column City.

Answer:

SELECT * FROM Players

ORDER BY Country, City

41 Consider a simple table Age as:

```
CREATE TABLE `doctor` (
  `dateOfBirth` datetime NOT NULL
);
```

Add a check constraint to make sure each doctor being entered in this DB is at least 24 years old! (YOU CANNOT USE Trigger)

Assume using MySQL DB engine

Answer:

CREATE TABLE doctors(

dateofBirth datetime NOT NULL

DATE CHECK (DATEDIFF(CURRENT_DATE(), dateofBirth) >= 8760)

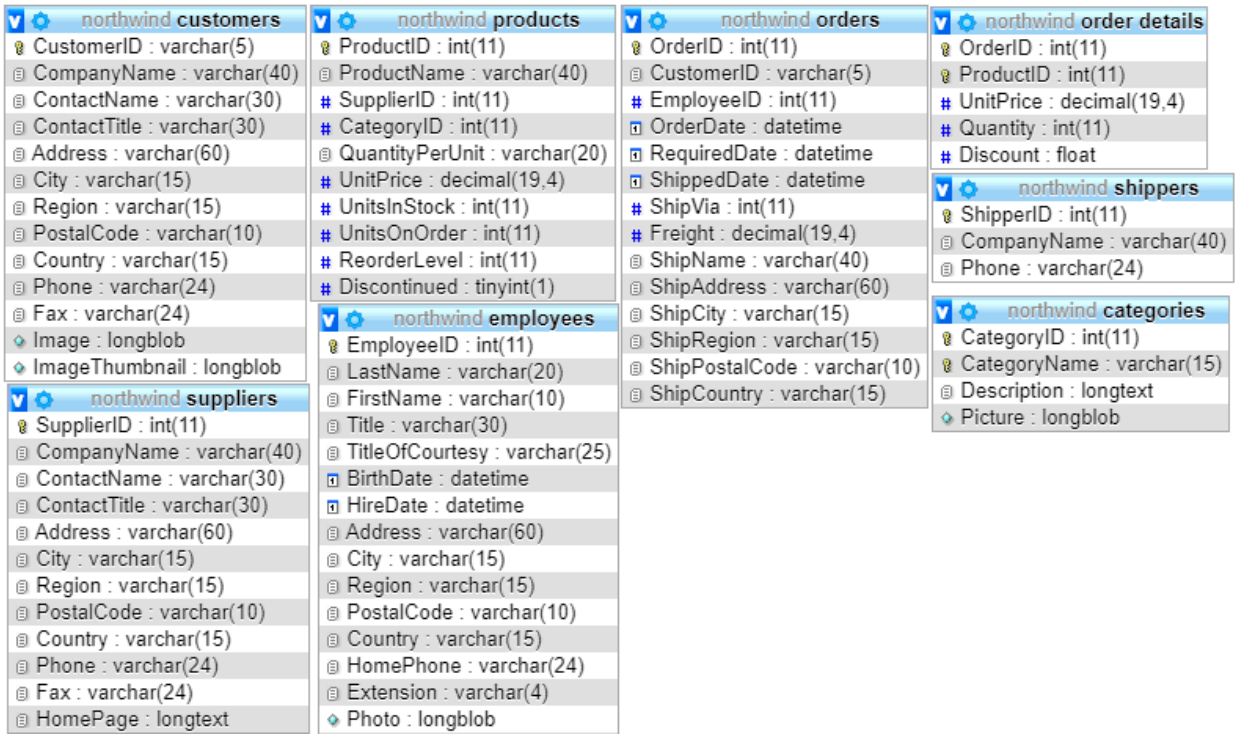
);

To see where 8760 is coming from, converted 24 years to days - which equals 8760 days

Second answer:

```
ALTER TABLE doctor ADD CHECK ( NOW()-dateOfBirth >= 240000000000 );
```

In all the questions below consider this database with the following tables also use SQLite



42 Write a SQL statement to list all customers and the length of their CustomerName (in MySQL)

Answer:
SELECT CustomerName, LENGTH(ContactName) AS LengthOfName
FROM Customers;

43 Write a SQL statement to list all customers whose name is longer than 20 characters

Answer:
SELECT CustomerName, LENGTH(CustomerName) AS LengthOfName
FROM Customers
where LENGTH(CustomerName)>20

44 Write a SQL statement to list all countries and their number of customers

Answer:
SELECT country, count(CustomerID) FROM [Customers] group by Country

45 Write a SQL statement to list all customers who live in Canada

Answer:
Answer in SQL Server and MySQL
SELECT * FROM [Customers] where LOWER(Country)= "canada"

46 Write a SQL statement to list all customers whose CustomerName starts with "a"

Answer:
SELECT * FROM [Customers] where CustomerName like "a%"

47 Write a SQL statement to list all customers who did not order anything

Answer: SELECT distinct customerName FROM [Customers], Orders where Customers.CustomerID not in (select CustomerID from orders)

48 Write a SQL statement to display the number of customers in the most populated country.

Answer: select max(c) from (SELECT count(customerID) as c, country FROM Customers group by country)

49 display the list of country (countries) with the most number of customers

Answer:
SELECT country, count(CustomerID) as cc FROM [Customers] group by country having cc in (select max(c) from (SELECT count(customerID) as c, country FROM Customers group by country)
)

50 How would you update the price of a product with product id 34 to 2.5?

Answer:
update products set price=2.5 where ProductID=34

51 How would you keep records of patients' visits to a clinic

Answer:
One working but not optimal solution would be this:

Patient name	Age	Clinic Visit
Sarah Brown	97	01/01/2002
John Smith	56	08/13/2018

52 Consider this table below, what could go wrong if a patient visits the office multiple times?

Patient name	Age	Clinic Visit
Sarah Brown	97	01/01/2002
John Smith	56	08/13/2018

Answer:
All repetition. For every visit we have to enter a new record which is fine but we have to enter the age of the patients for every visit too

53 we have come up with the new design below for keeping the records of patients' visits. What could go wrong ?

Patient name	Age	Clinic Visit 1	Clinic Visit 2
Sarah Brown	97	01/01/2002	
John Smith	56	08/13/2018	08/01/2019

Answer:
For the third visit we have to add a third column!
Sparse data! Some patients have only one visit, some patients have tens of visits. Then our DB will be filled with so much sparse data

54 list two major problems with the design below

Patient name	Age	Clinic Visit
Sarah Brown	97	01/01/2002
John Smith	56	08/13/2018
John Smith	56	08/01/2019

Answer:
1- every time we have to repeat the Age and Name of the patient
2- very error prone to human mistakes. Sometime names will be type upper case, sometime lower case

55 The DB below is not 1NF and 2NF normalized. How would you normalize it?

Patient name	Age	Clinic Visit 1	Clinic Visit 2
Sarah Brown	97	01/01/2002	
John Smith	56	08/13/2018	08/01/2019

Answer:

Patient

Patient_ID	Patient name	Age
1	Sarah Brown	97
2	John Smith	56

Visit

Visit_ID	Patient_ID	Clinic Visit
1	1	01/01/2002
2	2	08/13/2018
3	2	09/08/2019

56 What could be a potential problem with the way we designed the table below ?

Patient

Patient_ID	Patient name	Age
1	Sarah Brown	97
2	John Smith	56

Answer:
The way we stored Age is wrong! Every year we have to update the age

57 What is the proper way of storing the date of birth?

Answer:
`dateOfBirth` datetime

58 Develope the SQL statement to create the table below

Patient

Patient_ID	Patient name	Age
1	Sarah Brown	97
2	John Smith	56

Visit

Visit_ID	Patient_ID	Clinic Visit
1	1	01/01/2002
2	2	08/13/2018
3	2	09/08/2019

Answer:

```
CREATE TABLE `patient` (
  `patientid` int(11) NOT NULL,
  `name` varchar(100) DEFAULT NULL,
  `dateOfBirth` datetime NOT NULL,
  PRIMARY KEY(patientid)
);

CREATE TABLE `visit` (
  `visitid` int(11) NOT NULL,
  `patientid` int(11) NOT NULL,
  `visitDate` datetime NOT NULL,
  PRIMARY KEY(visitid),
  CONSTRAINT fk_has_patient FOREIGN KEY(patientid)
           REFERENCES patient(patientid)
);
```

59 How would you insert records in the tables patient and visit created in the previous question?

Answer:

```
INSERT INTO patient VALUES
(1, 'Sara Brown', '1901-01-01'),
(2, 'John Smith', '1941-01-01'),
```

```
(3, 'Jack Ma', '1961-01-30');

INSERT INTO visit VALUES
(1, 1, '2002-01-01'),
(2, 2, '2018-01-01'),
(3, 2, '2019-01-01'),
(4, 2, '2020-01-01');
```

60 In the table below, find the name of patients who were born before March 1st, 1940

patientid	name	dateOfBirth
1	Sara Brown	1901-01-01 00:00:00
2	John Smith	1941-01-01 00:00:00
3	Jack Ma	1961-01-30 00:00:00

Answer:
SELECT name FROM `patient` where dateOfBirth< CAST('1941-01-01' as DATE)

61 Without using CAST operator, in the table below, find the name of patients who were born before March 1st, 1940

patientid	name	dateOfBirth
1	Sara Brown	1901-01-01 00:00:00
2	John Smith	1941-01-01 00:00:00
3	Jack Ma	1961-01-30 00:00:00

Answer:
```
SELECT name FROM `patient` where dateOfBirth<'1940-03-01'
```

62 in the table below, find one of the oldest patient(s)

patientid	name	dateOfBirth
1	Sara Brown	1901-01-01 00:00:00
2	John Smith	1941-01-01 00:00:00
3	Jack Ma	1961-01-30 00:00:00

Answer:
In MySQL
SELECT * FROM `patient` order by dateOfBirth asc limit 1
In SQL server you need to use top instead of limit

63 In the table 'visit' below find the id of the patine who visited the office the most often

visitid	patientid	visitDate
1	1	2002-01-01 00:00:00
2	2	2018-01-01 00:00:00
3	2	2019-01-01 00:00:00
4	2	2020-01-01 00:00:00

Answer:

```
SELECT *, count(patientid) as c FROM `visit`group by patientid order by c DESC limit 1
```

Note: there might be more than one patient who visited the office the most often. This answer does not cover that case

64 What type of relationship do you see between the two tables below?

Answer:
1:M (one to many)

65 what data type would you choose to define the column ID in the table below?

Table1		Table2
id		id
2		'NULL'
2		2
3		3
NULL		
4		
NULL		

Note: there is no typos

Answer:
Because of 'NULL' which is a string, we can choose a datatype to store strings. That means the 1,2,3 will be treated as string.
Something like a fixed sized string such as CHAR(30) or variable sized string such as VARCHAR(30)

66 How would you create the tables of the previous questions?

Answer:

```
CREATE TABLE t1( id VARCHAR(30) );
```

```
CREATE TABLE t2( id VARCHAR(30) );
```

Note, we picked same data type for both table to be able to answer the subsequent questions

67 how would you insert the presented data in the two tables below?

Answer:

```
INSERT INTO t1 VALUES ('2'),('2'),('3'),(NULL),('4'),(NULL);
```

```
INSERT INTO t2 VALUES ('NULL'),(2),(3);
```

68 Using union operator, what would the following SQL statement return ?

```
SELECT * FROM t1 UNION SELECT * FROM t2;
```

Answer:

id
2
3
NULL
4
NULL

Note: as you see, 'NULL' and NULL appeared separately in the result because they are not the same

69 considering the tables of question 65, How many records whole the query below return?

Select * from t1, t2

Answer:
It generally returns as many rows as size of t1 X size of t2 which means 3x6 = 18
---

70 NULL Zip code

Suppose there is a table named Players in our DB with multiple columns including PlayerID (PRIMARY KEY), Country (of type CHAR(20)), City (of type CHAR(20)) and ZipCode (of type CHAR(6)) and weight (of type SMALLINT(3))

Write a Query to Select all records from the Players where the ZipCode column is NULL.

```
A) Select * From Players
Where ZipCode IS NULL

B) Select * From Players
Where ZipCode IS 'NULL'

C)Select * From Players
Where ZipCode IS not NULL

D) Select * From Players
Where ZipCode <> NULL
E) A and B
```

Answer: option A

71 in the table below we added the UNIQUE constraint. Was it a good idea?

```
ALTER TABLE `customers`
  ADD PRIMARY KEY (`CustomerID`),
   ADD UNIQUE KEY `CategoryName` (`CategoryName`);
```

Answer:
No, that means a specific city name can only appear in just one record

72 in the table t1 find all unique ID values which are not null

Table1

id
2
2
3
NULL
4
NULL

Answer:

```
Select DISTINCT * from t1 where id is not NULL
```

id
2
3
4

73 the table t2, the NULL value is represented as a string. Find all ID values which are not NULL

Table2

id
'NULL'
2
3

Answer:

```
Select DISTINCT * from t2 where id is not NULL
```

id
NULL
2
3

74 in the table below find all ID values which are not 'null'

Table2

id
'NULL'
2
3

Answer:

As you have noticed, the data in our table is upper case 'NULL' string, not lowercase 'null'. Also 'NULL' or 'null' are strings

```
Select DISTINCT * from t2 where id <> 'null'
```

Show all | Number of rows: 25

+ Options

id
2
3

Very interestingly, It did not return 'NULL' as part of the resulting table !
Why?

75 Why string comparisons are case insensitive in mySQL?

Answer: they are not! The reason is that when we created our database in phpMyAdmin, we used default character set

The default character set and collation are latin1 and latin1_swedish_ci, so nonbinary string comparisons are case insensitive by default.

76 how can we force the string comparisons to be case sensitive ?

Answer:
Using binary keyword :

```
SELECT * FROM `table` WHERE BINARY `column` = 'value'
```

77 find all ID values from the two tables below which are not NULL

Table1		Table2
id		id
2		'NULL'
2		2
3		3
NULL		
4		
NULL		

Note: there is no typos

Answer:

```
Select DISTINCT * from t1 where id is not NULL UNION Select DISTINCT * from t2 where id is not NULL
```

id
2
3
4
'NULL'

78 what would be the resulting table after execution of this query ?

```
SELECT T1.ID as T1ID, T2.ID as T2ID FROM T1 RIGHT JOIN T2 ON T1.id = T2.id
```

Table1		Table2
id		id
2		'NULL'
2		2
3		3
NULL		
4		
NULL		

Note: there is no typos

Answer:

T1ID	T2ID
2	2
2	2
3	3
NULL	'NULL'

79 what would the query below generate?

Table1		Table2
id		id
2		'NULL'
2		2
3		3
NULL		
4		
NULL		

Note: there is no typos

```
SELECT T2.ID as T2ID, T1.ID as T1ID FROM T2 RIGHT JOIN T1 ON T1.id = T2.id
```

Answer:

T2ID	T1ID
2	2
2	2
3	3
NULL	*NULL*
NULL	4
NULL	*NULL*

80: What kind of relationship do you see in the DB below ?

Note: The DB below has only one table named Boxers with a snapshot of its sample data (more records exist in this database that we don't see here)

Boxers

ID	Name	Weight	Trainer
1	**Emanuel Steward**	**150**	**null**
2	**Georges St. Pierre**	**180**	**Emanuel Steward**
3	**Jack Green**	**160**	**Emanuel Steward**
12	**John brown**	**155**	**Georges St. Pierre**
15	**...**	**...**	**...**

ID is a primary key of type INT
weight is also of type INT

Answer:

We can see that a trainer can train one or more boxers, but a boxer has only one trainer although not all boxers have a trainer (inferred from the table above). So, this can be a table with a 1:M recursive relationship

81: is this DB complying with 1NF and 2NF?

Answer:
It is 1NF but not 2NF.

82: write a SQL statement to create this table. Do you have to redesign the table's schema , or normalize it? If so, write your SQL statement to create the table which is normalized (1NF and 2NF) and can be used in next question

Note: it is ok to assume that after you redesigned your table, you again have to re-enter all records

Answer:

We have to modify Trainer to trainerID since Trainer refers to another boxer, who is actually uniquely identified by ID. This will also normalize the table to 2NF since TrainerName depends on ID of the trainer, not the ID of the boxer being trained (PK of table)

```
CREATE TABLE Boxers (

ID int NOT NULL,

Name varchar(255) NOT NULL,

Weight int NOT NULL,

TrainerID int,

PRIMARY KEY (ID),

FOREIGN KEY (TrainerID) REFERENCES Boxers(ID)

);

insert into boxers values (1,'Emanuel Steward',150,NULL),

(2,'Georges St. Pierre',180,1),

(3,'Jack GTreen',160,1),

(12,'John Brown',155,2)
```

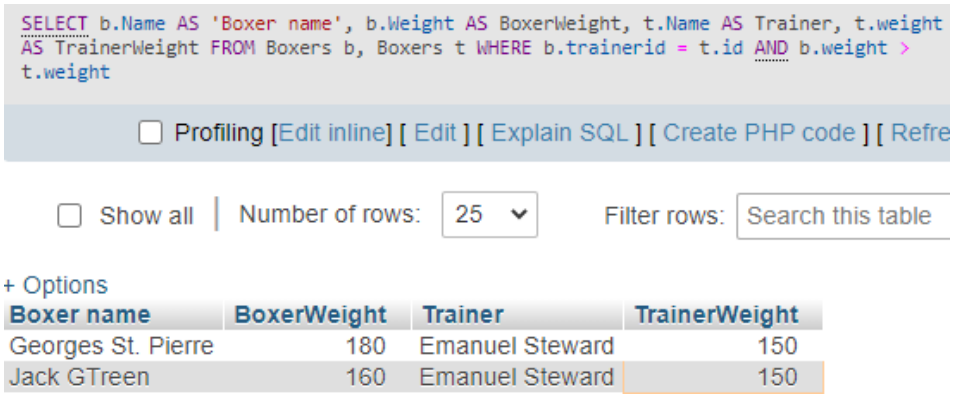
ID	Name	Weight	TrainerID
1	Emanuel Steward	150	*NULL*
2	Georges St. Pierre	180	1
3	Jack GTreen	160	1
12	John Brown	155	2

83 For the table you created in the previous question, write a query to list all boxers who are heavier than their trainer. Your resulting table must include:

Boxer name, BoxerWeigh, Trainer, TrainerWeight

Answer:

```
SELECT b.Name AS 'Boxer name', b.Weight AS BoxerWeight, t.Name AS Trainer, t.weight
AS TrainerWeight

FROM Boxers b, Boxers t

WHERE b.trainerid = t.id

AND b.weight > t.weight;
```



```
SELECT b.Name AS 'Boxer name', b.Weight AS BoxerWeight, t.Name AS Trainer, t.weight
AS TrainerWeight FROM Boxers b, Boxers t WHERE b.trainerid = t.id AND b.weight >
t.weight
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refre

Show all | Number of rows: 25 Filter rows: Search this table

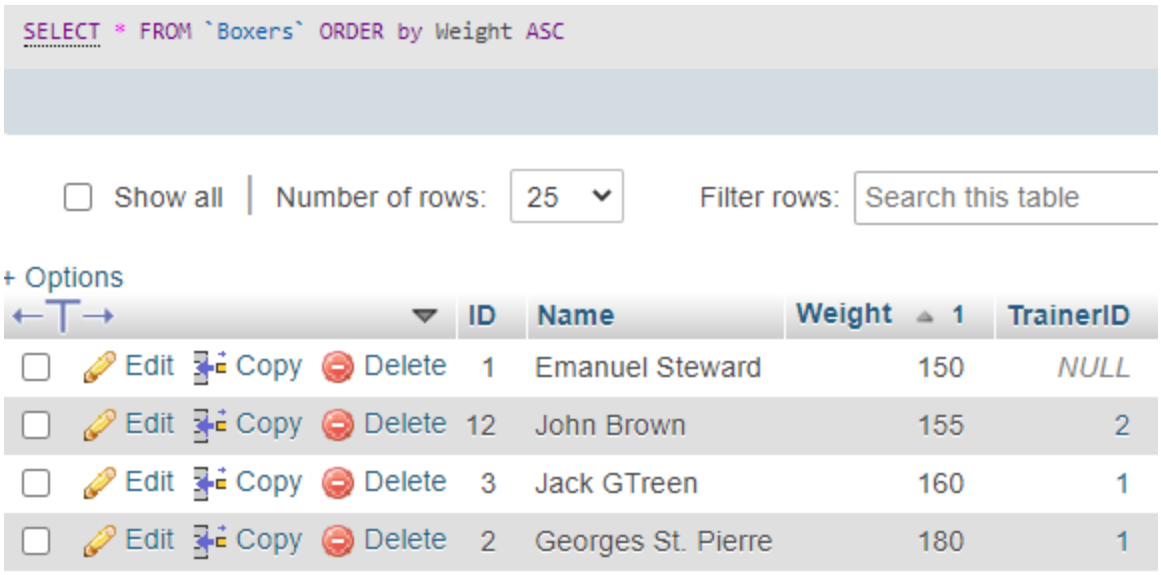
+ Options

Boxer name	BoxerWeight	Trainer	TrainerWeight
Georges St. Pierre	180	Emanuel Steward	150
Jack GTreen	160	Emanuel Steward	150

84: For the Boxers table, sort and list all the Boxers according to their weight in ascending order

Answer:

```
SELECT * FROM `Boxers` ORDER by Weight ASC
```



```
SELECT * FROM `Boxers` ORDER by Weight ASC
```

Show all | Number of rows: 25 Filter rows: Search this table

+ Options

	ID	Name	Weight ▲ 1	TrainerID
Edit Copy Delete	1	Emanuel Steward	150	*NULL*
Edit Copy Delete	12	John Brown	155	2
Edit Copy Delete	3	Jack GTreen	160	1
Edit Copy Delete	2	Georges St. Pierre	180	1

85 List all the boxers who had no trainer

Answer:

In MySQL:

```
SELECT * FROM `Boxers` WHERE TrainerID is NULL
```

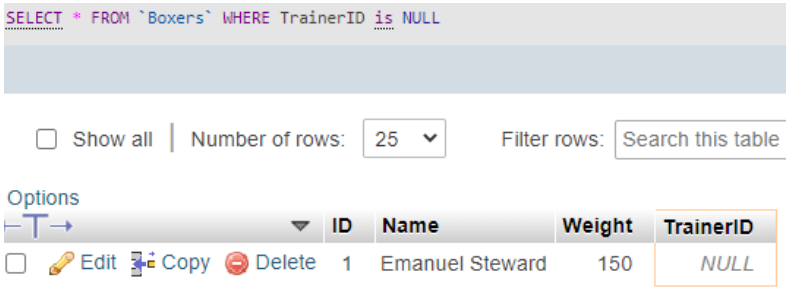
Note:

```
SELECT * FROM `Boxers` WHERE TrainerID = NULL
```

Will not work

```
SELECT * FROM `Boxers` WHERE TrainerID = 'NULL'
```

Will not work as 'NULL' is a string



86, normalize the table below, with some snapshot of entered data, and then list all pairs of boxers having the same trainer

Boxers

ID	Name	Weight	Trainer
1	**Emanuel Steward**	**150**	**null**
2	**Georges St. Pierre**	**180**	**Emanuel Steward**
3	**Jack Green**	**160**	**Emanuel Steward**
12	**John brown**	**155**	**Georges St. Pierre**
15	**...**	**...**	**...**

ID is a primary key of type INT

weight is also of type INT

Answer:

```
CREATE TABLE Boxers (

ID int NOT NULL,

Name varchar(255) NOT NULL,

Weight int NOT NULL,

TrainerID int,

PRIMARY KEY (ID),

FOREIGN KEY (TrainerID) REFERENCES Boxers(ID)

);

insert into boxers values (1,'Emanuel Steward',150,NULL),

(2,'Georges St. Pierre',180,1),

(3,'Jack GTreen',160,1),

(12,'John Brown',155,2)
```

The new table becomes:

ID	Name	Weight	TrainerID
1	Emanuel Steward	150	*NULL*
2	Georges St. Pierre	180	1
3	Jack GTreen	160	1
12	John Brown	155	2

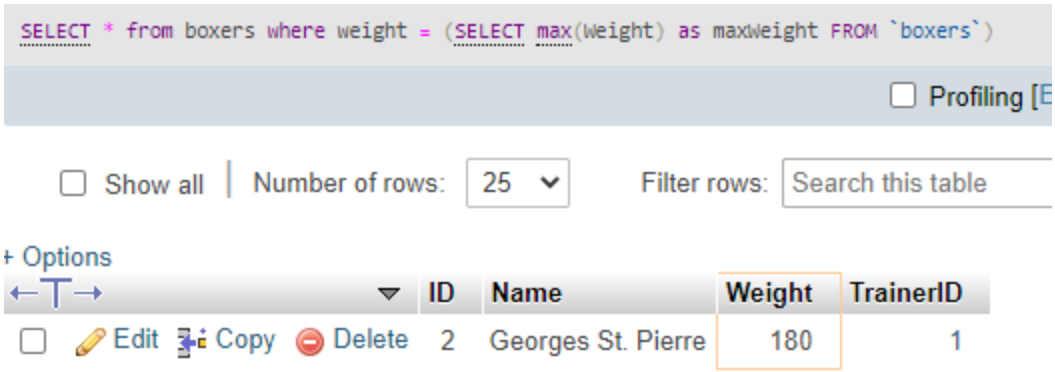
And now to find pairs of boxers with the same trainer:

87 For the recursive relation below, list the name of the heaviest boxer and their weight

ID	Name	Weight	TrainerID
1	Emanuel Steward	150	*NULL*
2	Georges St. Pierre	180	1
3	Jack GTreen	160	1
12	John Brown	155	2

Answer:

```
SELECT * from boxers where weight = (SELECT max(Weight) as maxWeight
FROM `boxers`)
```

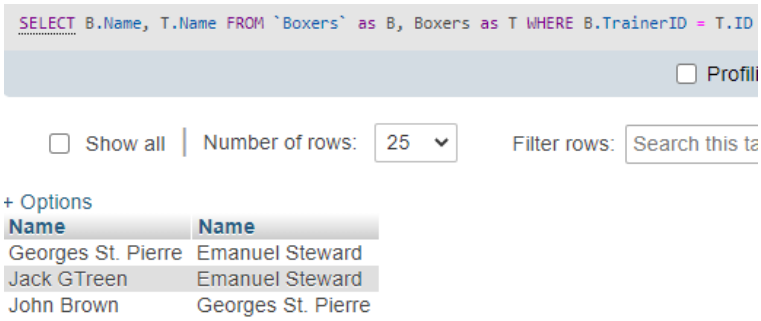


88 in the recursive relation below, list all boxers' names and their trainers' names

ID	Name	Weight	TrainerID
1	Emanuel Steward	150	*NULL*
2	Georges St. Pierre	180	1
3	Jack GTreen	160	1
12	John Brown	155	2

Answer:

```
SELECT B.Name, T.Name FROM `Boxers` as B, Boxers as T WHERE B.TrainerID = T.ID
```



89 List all boxers whose trainers' name start with 'G' (upper or lower case)

ID	Name	Weight	TrainerID
1	Emanuel Steward	150	*NULL*
2	Georges St. Pierre	180	1
3	Jack GTreen	160	1
12	John Brown	155	2

Answer:

```
SELECT B.Name, T.Name FROM `Boxers` as B, Boxers as T WHERE B.TrainerID = T.ID AND T.Name like "g%"
```

90 Write a query to list the first letter of all the boxers' names

ID	Name	Weight	TrainerID
1	Emanuel Steward	150	*NULL*
2	Georges St. Pierre	180	1
3	Jack GTreen	160	1
12	John Brown	155	2

Answer:

Its different in MySQL and SQL server

In SQL Server

```
select SUBSTR (Name, 1,1) from boxers
```

In MySQL

```
select left (Name, 1) from boxers
```

91 Finding pairs with some common properties in same table

Suppose there is a table named Items in our DB with multiple columns including

ItemID (PRIMARY KEY),

Name(of type CHAR(20))

and weight (of type SMALLINT(3))

Write a query that finds, for each Item A, another Item B that has the same weight. Find all such pairs of Items (A B) and against each pair, the common weight. The query should thus have three columns. Order the results by the common weight in ascending order

your result set must have three columns

Item A, Item B, commonWeight

Answer:

assuming the name of the table is "Tab" :

```
SELECT a.name AS 'Item A', b.name AS 'Item B', a.weight AS commonWeight

from Tab  AS a, Tab AS b

where a.itemid < b.itemid

and a.weight = b.weight

order by commonWeight;
```

92 Query optimization

Suppose there is a table named Items in our DB with multiple columns including

ItemID (PRIMARY KEY),

Name(of type CHAR(20))

and weight (of type SMALLINT(3))

How would you optimize this query statement?

SELECT COUNT(DISTINCT ItemID) FROM Items WHERE Name LIKE 'a%';

Note: Wrong suggestions result in deduction of mark

Answer:

Since ItemID is the primary key, the value for each row should be unique so you don't need **DISTINCT** in the statement. Change it to:

SELECT COUNT(ItemID) FROM Items WHERE Name LIKE 'a%';

93 How would you create the tables below ?

Consider the tables you created for the next few questions.

Table1		Table2
id		id
222		5
222		222
300		300
5		
4		
NULL		

Answer:

```
CREATE TABLE table1 ( id int );

INSERT INTO table1 VALUES (222),(222),(300),(5),(4),(NULL);

CREATE TABLE table2 ( id int );

INSERT INTO table1 VALUES (5),(222),(300);
```

94 Using the previous tables Table1, Table2, What would be the result of the following MySQL statement:

```
SELECT * FROM Table1 INNER JOIN Table2 ON table1.id =
table2.id;
```

Answer:

id	id
222	222
222	222
300	300
5	5

95 Using UNION , what would be the result of the following MySQL statement:

```
SELECT * FROM table1 UNION SELECT * FROM table2;
```

Answer:

id
222
300
5
4
NULL

96 Using RIGHT JOIN, what would be the result of the following MySQL statement:

```
SELECT * FROM Table1 RIGHT JOIN Table2 ON table1.id =
table2.id;
```

Answer:

id	id
222	222
222	222
300	300
5	5

97 Write a query in MySQL to create a table with the following fields which is part of a DB designed for a theme park

childID: of type character of length 100 and has to be unique and to be set as primary key

name: of type variable character of length up to 20 and its ok to be null

height (in cm): of type decimal numbers which can be as large as 299.99 (Example of valid numbers: 120.99, 100.990)

height cannot be null

In your SQL statement, you have to make sure height cannot be entered less than 100 and more than 299.99

Answer:

Assuming I can name the table to "Children":

CREATE TABLE Children (

childID char(100) NOT NULL,

name varchar(20),

`height (in cm)` decimal (5,2) NOT NULL,

PRIMARY KEY (childID),

CONSTRAINT minheight_chk CHECK(`height (in cm)` >= 100),

CONSTRAINT maxheight_chk CHECK(`height (in cm)` <= 299.99));

Marking (3 points)

1 DECIMAL(5,2) NOT NULL

DECIMAL(6,3) NOT NULL

2

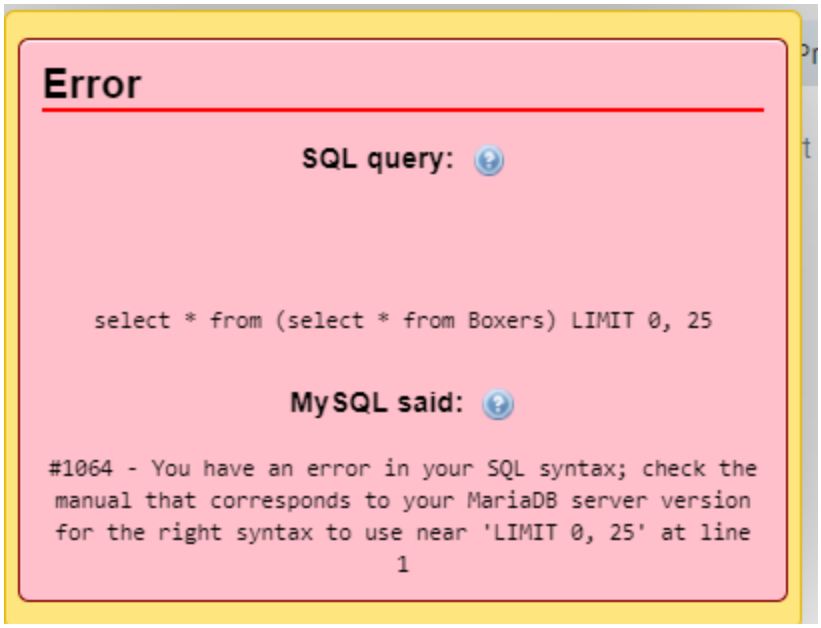
CONSTRAINT minheight_chk CHECK(`height (in cm)` >= 100),

CONSTRAINT maxheight_chk CHECK(`height (in cm)` <= 299.99));

98 Why this Syntax error is thrown in phpMyAdmin after running the query for the table below?

ID	Name	Weight	TrainerID
1	Emanuel Steward	150	*NULL*
2	Georges St. Pierre	180	1
3	Jack GTreen	160	1
12	John Brown	155	2

```
select * from (select * from Boxers)
```



Answer:
Because we need to add an alias to the inner table: select * from (select * from Boxers) as T
The question is why do we need to do that?

because it is the syntax of SQL (and other DBMS) if we add a LIMIT clause at the end.
The problem is that the LIMIT clause is automatically being added by phpMyAdmin!! (see the attached image for error message)

99 How can we count the number of records existing in a table?

Answer:
Simply by counting the number of primary keys.
For example:

```
SELECT COUNT(ID) FROM Table
```

100 For the table below, list the tainer(s) with the most number of trainees

ID	Name	Weight	TrainerID
1	Emanuel Steward	150	*NULL*
2	Georges St. Pierre	180	1
3	Jack GTreen	160	1
12	John Brown	155	2

Answer:
The tricky part is that we cannot assume there is only one record corresponding to the max values .

So lets solve this exercise in a few steps.

First: lets see how many trainees each trainer has:

```
SELECT TrainerID, COUNT(ID) as traineeCount FROM `Boxers` group by TrainerID
```

```
SELECT TrainerID, COUNT(ID) as traineeCount FROM `Boxers` group by TrainerID
```

Show all | Number of rows: 25 Filter rows: Search this

- Options

TrainerID	traineeCount
NULL	1
1	2
2	1

```
Now we get the max value of traineeCount
```

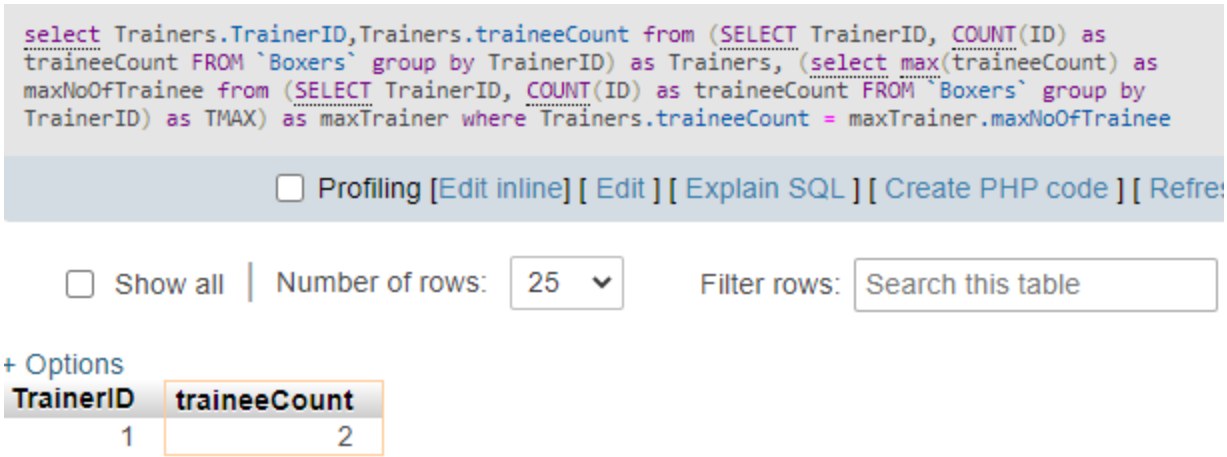
```
select max(traineeCount) from (SELECT TrainerID, COUNT(ID) as traineeCount FROM `Boxers` group by TrainerID) as TMAX
```

max(traineeCount)
2

Now, lets relate the tranerIDs with their names

It will be so hard as the top row has trainerID of null! Here we shold have set business logic so that only one trainee can has trainerID of null, which means only one trainer could be the originating trainer who started training otherds and himself had no triner

```
select Trainers.TrainerID,Trainers.traineeCount
from
(SELECT TrainerID, COUNT(ID) as traineeCount FROM `Boxers` group by TrainerID)
as Trainers,
(select max(traineeCount) as maxNoOfTrainee from (SELECT TrainerID, COUNT(ID) as
traineeCount FROM `Boxers` group by TrainerID) as TMAX)
as maxTrainer
where Trainers.traineeCount = maxTrainer.maxNoOfTrainee
```



By now we only got the ID of the trainer, not their name! So lets so another subquery to fetch their names too:
So here is the final answer:

```
select name, traineeCount
from
  Boxers,
    (select Trainers.TrainerID,Trainers.traineeCount
    from
    (SELECT TrainerID, COUNT(ID) as traineeCount FROM `Boxers` group by TrainerID)
    as Trainers,
```

```
    (select max(traineeCount) as maxNoOfTrainee from (SELECT TrainerID, COUNT(ID) as
traineeCount FROM `Boxers` group by TrainerID) as TMAX)
    as maxTrainer
    where Trainers.traineeCount = maxTrainer.maxNoOfTrainee)
  as TmaxTrainers

WHERE Boxers.ID = TmaxTrainers.TrainerID
```

Prepared by Dr. Amir Amintabar,

Open Education Project at BCIT.ca

Ver. 2023-07-01