The Solution For The Tasks

Aleksandr Kasian

05-27-2020

Contents

Solution of the Problem 1

Recall the Task

Rambler Group's advertising campaign uses most fascinating and most memorable banners. Analytics have access to the databases containing data regard the banner' showing. The **Shows_table** contains:

- *show_id* - an identifier of a showing
- *day* - a day of a showig

show_id	day
12367	2018-10-04
28736	2019-02-22
19862	2019-01-31

The **Click_table** contains:

- *click_id* - a show identifier clicked by an user
- *bounce* - an user dismissing from an advertising after click (0 - when an user relinked to the site he keened in the information on the site. 1 - an user immediatly left the site.)

click_id	bounce
12367	1
15627	0
28735	0

You need to get all users who clicked at a banner in February 2020, and they din't reject an advertising.

Step-by-Step Solution

1. First of all, I'd like to obtain all users, I mean the whole database of users, without any filters. But I baffled by the task, because it is clearly said that we need to fetch the "**users**", however I do not really aware what exactly is the "**users**". Ensuing of this I decided to derive the *show_id* (it also can be the *click_id*, result will be the same).

```sql
SELECT show_id FROM Shows_table
```

2. The next scanty modification is to retrieve unique *show_id*. For instance, if an user clicked twice on the same banner, it shouldn't be represented twice, therefore distinct it.

```sql
SELECT DISTINCT show_id FROM Shows_table
```

3. The next step is to detect those who clicked and remained on the advertising site. The *INER JOIN* the most appropriating command for this purpose.

   The INNER JOIN keyword selects records that have matching values in both tables.

```
SELECT DISTINCT show_id FROM Shows_table
INNER JOIN Clicks_table ON
show_id=click_id AND bounce='0'
```

   Let's deem this query on the example.

show_id	day
12367	2018-10-04
15627	2020-02-22
28736	2019-01-31

click_id	bounce
12367	1
15627	0
28736	0

   In the above case the banners which ids are **15627** and **28735** are those banners that an user remained by clicking on. So, this request returns such list:

```
SELECT DISTINCT show_id FROM Shows_table
```

12367
15627
28736

   If be honest this one does the same as the previous one.

```
SELECT DISTINCT show_id FROM Shows_table
INNER JOIN Clicks_table ON
show_id=click_id
```

   And the last thing in this subsection is to filter the case when an user remains on the advertising site. According the task "*bounce - an user dismissing from an advertising after click (0 - when an user relinked to the site he keened in the information on the site. 1 - an user immediatly left the site.)*" This request perfectly does this, and returns such id's list.

```
SELECT DISTINCT show_id FROM Shows_table
INNER JOIN Clicks_table ON
show_id=click_id AND bounce='0'
```

28736
15627

4. The last what we should to do is to distill by date. The task says "*You need to get all users who clicked at a banner in February 2020, and they din't reject an advertising.*", it does pretty simple in SQL, by adding up this to the request.

```
... day BETWEEN '2020-02-01' AND '2020-02-29'
```

5. The result looks like this:

```
SELECT DISTINCT show_id FROM Shows_table
INNER JOIN Clicks_table ON
show_id=click_id AND bounce='0' AND day BETWEEN '2020-02-01' AND '2020-02-29'
```

   In our example this query has to return only the **15627**, because this is the only *click_id* within February 2020.

Testing

In purpose to test my solution I was using the SQLFiddle and PostgreSQL 9.6.

1. I created the test-tables, using DLL, by these queries:

```
CREATE TABLE Shows_table (
    show_id INT,
    day DATE
);
```

```
CREATE TABLE Clicks_table (
    click_id INT,
    bounce BOOLEAN
);
```

2. Fill them (tables) up, by using these requests

```
INSERT INTO Shows_table (show_id, day) VALUES (12367, '2018-10-04');
INSERT INTO Shows_table (show_id, day) VALUES (28736, '2019-02-22');
INSERT INTO Shows_table (show_id, day) VALUES (19862, '2019-01-31');
INSERT INTO Shows_table (show_id, day) VALUES (11111, '2020-02-04');
INSERT INTO Shows_table (show_id, day) VALUES (22222, '2020-02-01');
INSERT INTO Shows_table (show_id, day) VALUES (33333, '2020-02-29');
INSERT INTO Clicks_table (click_id, bounce) VALUES (12367, '1');
INSERT INTO Clicks_table (click_id, bounce) VALUES (28736, '0');
INSERT INTO Clicks_table (click_id, bounce) VALUES (19862, '0');
INSERT INTO Clicks_table (click_id, bounce) VALUES (11111, '1');
INSERT INTO Clicks_table (click_id, bounce) VALUES (22222, '0');
INSERT INTO Clicks_table (click_id, bounce) VALUES (33333, '0');
```

And now the tables looks like these:

show_id	day
12367	2018-10-04
28736	2019-02-22
19862	2019-01-31
11111	2020-02-04
22222	2020-02-01
33333	2020-02-29

click_id	bounce
12367	1
28736	0
19862	0
11111	1
22222	0
33333	0

3. For this example the query (my solution) returns the **22222** and **33333**.

   - **12367** - isn't suitable, because an user left the site and it was in October 2018
   - **28736** - isn't suitable, because it was in February 2019.
   - **19862** - isn't suitable, because it was in January 2019.
   - **11111** - isn't suitable, because an user left the site.

Answer

The answer to the **Problem 1**

```
    SELECT DISTINCT show_id FROM Shows_table
    INNER JOIN Clicks_table ON
    show_id=click_id AND bounce='0' AND day BETWEEN '2020-02-01' AND '2020-02-29'
```

Solution of the Problem 2

Recall the Task

The friendly Rambler Group's community likes to play in the table football: At the odd days they play before lunch, at the even days the play after lunch. They are splitting at the N teams among each other, and every team plays with each another team. Because of the splitting onto the teams is randomly, the product of the games is random. Also I would note that there are no ties. Only win or lose.

1. Estimate the probability if one of the teams will finish the tournament without defeat.
2. How many times do you need to hold a tournament, so that with a probability of 98% at least once this happened?

Solution for the first question

What we have:
N - number of teams P (win) = 1/2 P (lose) = 1/2
Necessary to seek:
P (if one of the teams will finish the tournament without defeat) - ?

There are two ways to solve it, the first one is the combinatorics and the second is the probability theory. I will show the each one.

Combinatorics way

1. First of all we need to estimate how many games N teams plays. Because of each team plays with every another team, therefore the number of games is the number of combination N by 2. ("*by 2*" because two teams participate in a game.)

I'd like to recall that the factorial of a positive integer n, denoted by n!, is the product of all positive integers less than or equal to n. *For instance, the factorial of N is* $1 \times 2 \times 3 \times ... \times (N-2) \times (N-1) \times N$.

   .

   $$\binom{N}{2} = \frac{N!}{2! \times (N-2)!} = \frac{\cancel{1 \times 2 \times 3 \times ... \times (N-2)} \times (N-1) \times N}{2 \times \cancel{1 \times 2 \times 3 \times ... \times (N-2)}} = \frac{(N-1) \times N}{2}$$

2. Each team either wins nor loses, therefore there are two outcomes. Ensuing of this the amount of all possible outcomes in whole tournament is $2^{\frac{(N-1) \times N}{2}}$.

3. Let's assume that the team A is won each game in the tournament. It means that from the *(N - 1)* games the team A is won *(N - 1)* games. Moreover, it means that the quantity of uncertain game's outcomes declines on *(N - 1)*. Therefore the overall number of outcomes, with the condition that team A wins all games is:

   $$2^{\frac{(N-1) \times N}{2} - (N-1)}$$

4. However, nobody knows which exactly team wins all games. It's not necessary that team A wins, as same as A it could be either team B, nor team C, nor team D and so on. If we will deem each case (for each team), then the overall amount of outcomes increases by N (number of teams) times.

$$\underbrace{2^{\frac{(N-1)\times N}{2}-(N-1)}}_{\text{if team A wins}}+\underbrace{2^{\frac{(N-1)\times N}{2}-(N-1)}}_{\text{OR team B wins}}+\underbrace{2^{\frac{(N-1)\times N}{2}-(N-1)}}_{\text{OR team C wins}}+\cdots+\underbrace{2^{\frac{(N-1)\times N}{2}-(N-1)}}_{\text{OR team N wins}}$$

For N teams it (number of all possible outcomes, where one of the teams wins) equals to:

$$N \times 2^{\frac{(N-1)\times N}{2}-(N-1)}$$

5. Recall that probability definition:

$$\boxed{\text{\textit{The probability of an event is a number indicating how likely that event will occur. And the probability of an event is: } } P(E)=\frac{m}{N}, \text{\textit{ where m - the number of demanded outcomes and N - the number of all possible outcomes.}}}$$

Therefore to obtain the probability of a team will finish the tournament without defeat we need to divide the number of outcomes where one of the teams is won whole tournament ($N \times 2^{\frac{(N-1)\times N}{2}-(N-1)}$) by number of all possible outcomes ($2^{\frac{(N-1)\times N}{2}}$):

$$P(E)=\frac{N \times 2^{\frac{(N-1)\times N}{2}-(N-1)}}{2^{\frac{(N-1)\times N}{2}}}=N \times 2^{\frac{(N-1)\times N}{2}-(N-1)-\frac{(N-1)\times N}{2}}=N \times 2^{-(N-1)}=\frac{N}{2^{N-1}}$$

6. Here it is, the answer. It means that the probability of a team wins the whole tournament without lose is $\frac{N}{2^{N-1}}$.

Probability theory way

If be honest this approach much easier than previous one.

1. Let's assume that the team A wins each game in the tournament, and the probability of this event is $(\frac{1}{2})^{N-1}$. $\frac{1}{2}$ - the probability of an outcome (whether lose nor victory) of one game. And $(N-1)$ the number of games which plays the team A.

$$\underbrace{\frac{1}{2}}_{\text{A wins 1-st game}} \times \underbrace{\frac{1}{2}}_{\text{AND A wins 2-nd game}} \times \underbrace{\frac{1}{2}}_{\text{AND A wins 3-d game}} \times \cdots \times \underbrace{\frac{1}{2}}_{\text{AND A wins (N - 1)-th game}}$$

2. However the probability of that the **one** of the teams wins the tournament is

$$P(\text{team A wins})+P(\text{team B wins})+P(\text{team C wins})+\cdots+P(\text{team N wins})=$$

$$=\underbrace{\left(\frac{1}{2}\right)^{N-1}+\left(\frac{1}{2}\right)^{N-1}+\left(\frac{1}{2}\right)^{N-1}+\cdots+\left(\frac{1}{2}\right)^{N-1}}_{\text{N times}}=N \times \left(\frac{1}{2}\right)^{N-1}=\frac{N}{2^{N-1}}$$

3. So, the probability that one of the teams wins the tournament without defeat is

$$P(E)=\frac{N}{2^{N-1}}$$

Solution of the second part

Let's recall the question:
How many times do you need to hold a tournament, so that with a probability of 98% at least once this happened (one of the teams wins the tournament without defeat)?

1. Let x be the quantity of tournaments required to befell to approximate the probability of the event (victory without defeat) to 98%.

2. Then,

$$P(\text{it happens at least once}) = 1 - P(\text{it never happens}) = 1 - (1 - N \times 2^{1-N})^x$$

   Let's explore the equation above. In the previous part we've figured out the probability that one of the teams wins the whole tournament without defeat is $\frac{N}{2^{N-1}}$ or $\left(N \times 2^{1-N}\right)$. Of course the probability of that this event never happens is *(100% - probability of happens at least once)*. Therefore, in our case it looks like $1 - \left(N \times 2^{1-N}\right)$, in other words this equation shows us the probability that there is no a team which wins a tournament without defeat. But this probability represents the only tournament, however we have x tournaments, and in the each one there is shouldn't be a team which wins at least 1 tournament without defeat.

$$P(\text{it never happens}) =$$
$$\underbrace{\underbrace{1 - \left(N \times 2^{1-N}\right)}_{\text{NOT today}} \times \underbrace{1 - \left(N \times 2^{1-N}\right)}_{\text{AND NOT tomorrow}} \times \cdots \times \underbrace{1 - \left(N \times 2^{1-N}\right)}_{\text{AND NOT after x days}}}_{\text{x times}} = \left(1 - N \times 2^{1-N}\right)^x$$

   And the last thing, when we subtracting from 100% the probability of the event when there is no a team which wins at least one tournament without defeat. This shows us the probability when a team which wins at least one tournament exsists.

3. And according the task's condition this probability equal to 98% or 0.98.
   - $P(\text{it happens at least once}) = 1 - P(\text{it never happens}) = 0.98 \implies$
   - $P(\text{it never happens}) = \left(1 - N \times 2^{1-N}\right)^x = 0.02$

4. Substitute it under logarithm

$$\log\left(\left(1 - N \times 2^{1-N}\right)^x\right) = \log 0.02$$

5. And use log-power rule

$$x \times \log\left(1 - N \times 2^{1-N}\right) = \log 0.02$$

6. Derive the x

$$x = \frac{\log 0.02}{\log\left(1 - N \times 2^{1-N}\right)}$$

   Here is it! The number of required tournaments is $\frac{\log 0.02}{\log(1 - N \times 2^{1-N})}$. Of course it depends on number of teams, for instance for two teams the only tournament enough.

Answer

The answer for the **Problem 2**

1. The probability that one of the teams wins the tournament without defeat is $\frac{N}{2^{N-1}}$
2. The number of required to befell tournaments to approximates the probability of the event (victory without defeat) to 98% is $\frac{\log 0.02}{\log(1-N\times(2)^{1-N})}$