

## **Brief about project idea:**

Kickstarter is one of the most popular crowdfunding platforms on the internet. The aim of this project is to predict the success or failure of a Kickstarter campaign at launch time.

Crowdfunding is the practice of funding a project or venture by raising monetary contributions from many people. The majority of today's crowdfunding happens online through various websites and one of the most prominent is Kickstarter.

The steps to start a Kickstarter project are; start a campaign, set the minimum funding goal, set reward levels, and choose a deadline. The most important aspect to know about launching a Kickstarter project is that if the project falls short of meeting its minimum funding goal, the project will not receive any fund. The projects analyzed in this project fall into one of 14 categories (Art, Comics, Dance, Design, Fashion, Film & Video, Film & amp; Video, Food, Games, Music, Photography, Publishing, Technology, Theater) and 51 subcategories. Only 55% of campaigns reach their funding goal thus it is extremely important for creators to know the factor(s) that might impact the outcome of their project before launch.

This project will take inputs from users using website and machine learning algorithms will provide various prediction / recommendations which are helpful to conduct the crowdfunding project.

Input from the Users on Website / Predictor for ML Algorithm

- Category and Subcategory of Project
- Location of the Project (City and State)
- Goal in Dollars
- Levels, Duration and No. of Update for the Project

## Conceptual Schema:

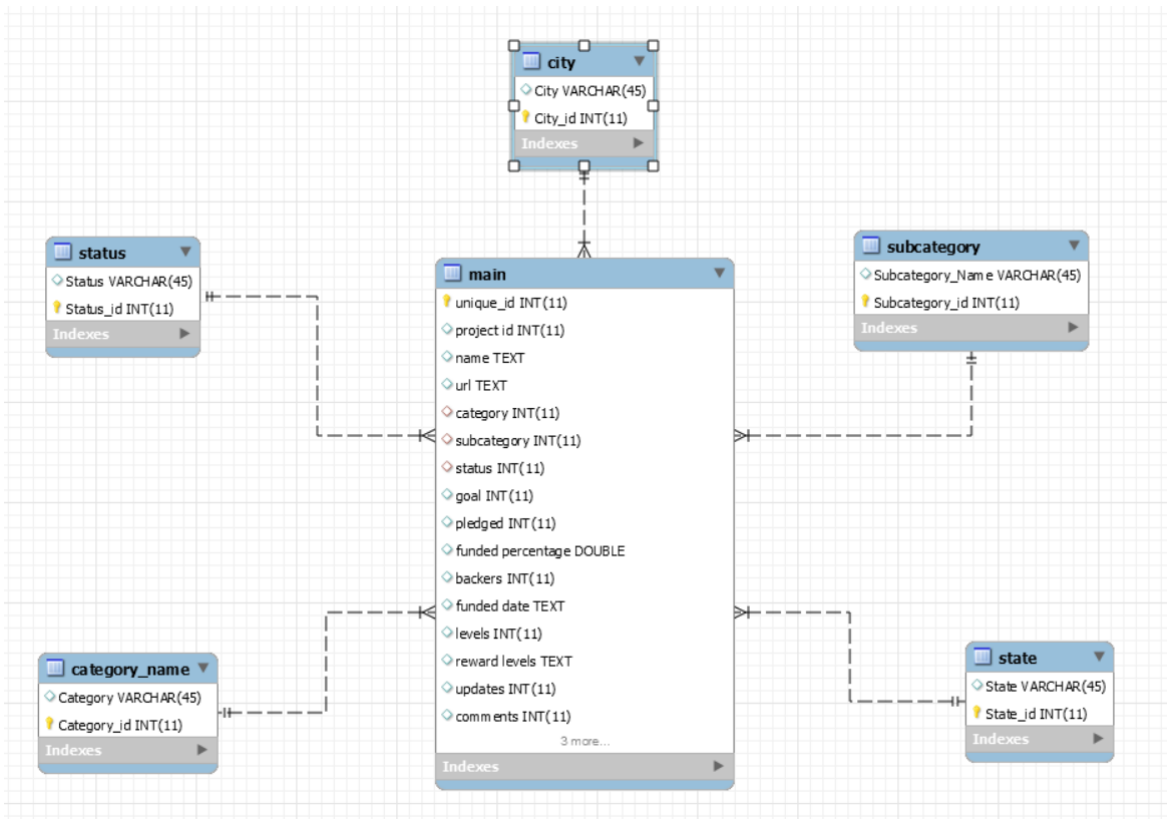
The database consists of 6 tables which are in 3<sup>rd</sup> normal form. The main table consist of all the project details all the details entered by the user to estimate the success or failure of a project.

The smaller tables consist of the integer values assigned to the categorical variables which are then used to build the random forest method to predict the classification of the project in successful or failed project list.

The data set consist of 46000 rows and 16 attributes or features.

About the table:

1. City: Primary Key: City\_id
2. State: Primary Key: State\_id
3. Subcategory: Primary Key: Subcategory\_id
4. Category: Primary Key: Category\_id
5. Status: Primary Key: Status\_id
6. Main:
  - a. Primary Key: unique\_id
  - b. Foreign Key: city, state, category, subcategory, status



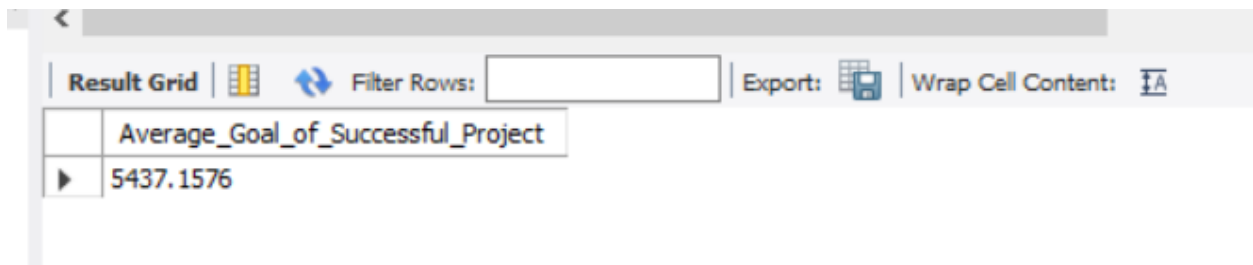
# Exploration of the database: SQL Queries

#1 Average goal of successful and failed projects

Query 1:

```
select avg(goal) as Average_Goal_of_Successful_Project from main
where status = 1;
```

Output:



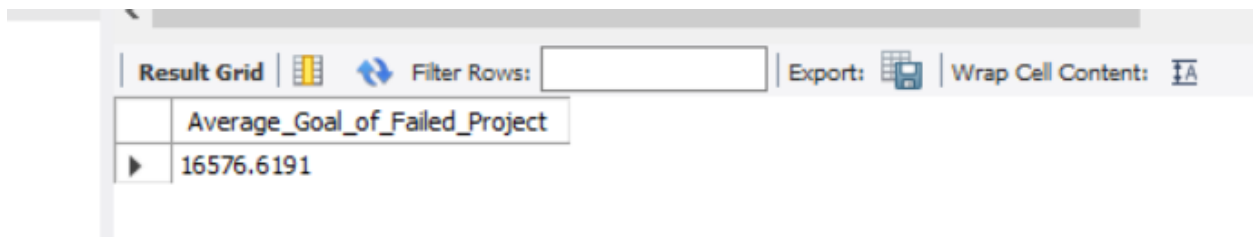
The screenshot shows a database interface with a 'Result Grid' tab. The grid has one column titled 'Average\_Goal\_of\_Successful\_Project' and one row with the value '5437.1576'. Above the grid is a 'Filter Rows' input field and buttons for 'Export' and 'Wrap Cell Content'.

Average_Goal_of_Successful_Project
5437.1576

Query 2:

```
select avg(goal) as Average_Goal_of_Failed_Project from main
where status = 0;
```

Output:



The screenshot shows a database interface with a 'Result Grid' tab. The grid has one column titled 'Average\_Goal\_of\_Failed\_Project' and one row with the value '16576.6191'. Above the grid is a 'Filter Rows' input field and buttons for 'Export' and 'Wrap Cell Content'.

Average_Goal_of_Failed_Project
16576.6191

#2 Success Percentage according to category

Query 3:

```
select Category , avg(status)*100 as percentage from main
group by category;
```

Output:

Category	percentage
0	56.3592
1	54.1073
2	76.1171
3	46.0000
4	32.5128
5	53.2847
6	50.9931
7	50.4425
8	43.2234
9	67.8104
10	43.4234
11	39.3065
12	38.3792

#3 Total funding in a state

Query 4:

```
select state.state, sum(goal) from main left join state on main.state = state.State_id
where status = 1
group by state
order by sum(goal) desc;
```

Output:

state	sum(goal)
CA	30016307
OH	25352629
UT	5407033
WI	4555363
IL	4144059
MA	4002446
PA	3377670
TX	3315798
RI	3184646

#### #4 Total Extra Funding

Query 5:

```
select sum(pledged-goal) as Extra_Funding from main
where status = 1;
```

Output:

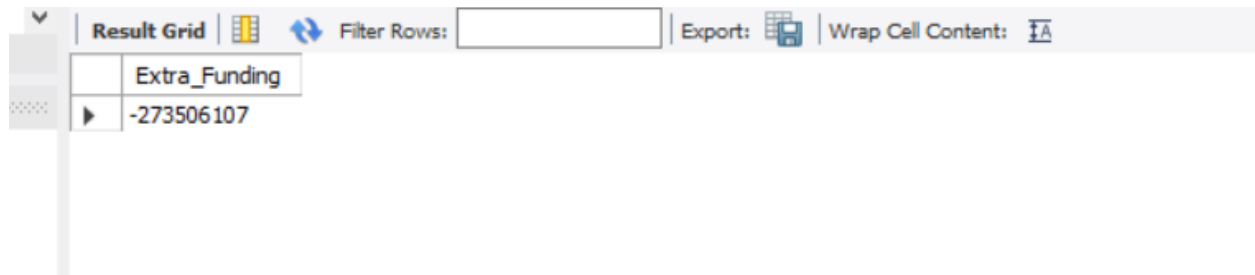


Extra_Funding
68645644

Query 6:

```
select sum(pledged-goal) as Extra_Funding from main
where status = 0;
```

Output:



Extra_Funding
-273506107

#### #5 Trending subcategory

Query 7:

```
select subcategory_name, sum(backers) as Trending
from main left join subcategory on main.subcategory = subcategory.subcategory_id
group by subcategory
order by sum(backers) desc;
```

Output:

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	subcategory_name	Trending			
▶	Video Games	356565			
	Product Design	291436			
	Documentary	233733			
	Music	191544			
	Short Film	120930			
	Indie Rock	97638			
	Theater	95249			
	Comics	90702			
	Film & TV: Video	88007			

#6 Average goal of successful and failed projects

Query 8:

```
select avg(goal) from main
where status = 1;
```

Query 9:

```
select avg(goal) from main
where status = 0;
```

Output:

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	avg(goal)				
▶	16576.6191				

#8 Total Projects for success and failure

Query 10:

```
select count(unique_id) from main
where status = 1;
```

Output:



The screenshot shows a database interface with a toolbar at the top containing 'Result Grid', 'Filter Rows', 'Export', and 'Wrap Cell Content' options. Below the toolbar is a table with one column and one row. The column header is 'count(unique\_id)' and the value in the row is '21066'.

count(unique_id)
21066

Query 11:

```
select avg(unique_id) from main  
where status = 0;
```

Output:



The screenshot shows a database interface with a toolbar at the top containing 'Result Grid', 'Filter Rows', 'Export', and 'Wrap Cell Content' options. Below the toolbar is a table with one column and one row. The column header is 'avg(unique\_id)' and the value in the row is '23008.2666'.

avg(unique_id)
23008.2666

#9 Average Updates

Query 12:

```
select avg(duration) from main  
where status = 1;
```

Output:



The screenshot shows a database interface with a toolbar at the top containing 'Result Grid', 'Filter Rows', 'Export', and 'Wrap Cell Content' options. Below the toolbar is a table with one column and one row. The column header is 'avg(duration)' and the value in the row is '37.55090619956317'.

avg(duration)
37.55090619956317

Query 13:

```
select avg(duration) from main  
where status = 0;
```

Output:

▼	Result Grid	Filter Rows:	Export:	Wrap Cell Content:
	avg(duration)			
▶	42.52084418938273			

#10

Query 14 with function:

```
select name, backers_amount(backers) from main;
```

Output:

▼	Result Grid	Filter Rows:	Export:	Wrap Cell Content:	Fetch ro
	name	goal_status(goal)	status		
▶	WHILE THE TREES SLEEP	High	1		
	Educational Online Trading Card Game	Low	0		
	GETTING OVER - One son's search to finally kno...	High	1		
	The Launch of FlyeGrlRoyalty &quot;The New N...	Low	0		
	Dinner Party - a short film about friendship... a...	Low	1		
	Mezzo	Low	0		
	Help APORTA continue to make handwoven/knit...	Low	1		
	Music - Comedy - Album!	Low	1		

#11

Query 15 with function:

```
select name, goal_status(goal), status from main;
```

Output:

▼	Result Grid	Filter Rows:	Export:	Wrap Cell Content:	Fetch rows:
	name	level_amount(levels)	status		
▶	WHILE THE TREES SLEEP	Low	1		
	Educational Online Trading Card Game	Low	0		
	GETTING OVER - One son's search to finally kno...	High	1		
	The Launch of FlyeGrlRoyalty &quot;The New N...	Low	0		
	Dinner Party - a short film about friendship... a...	Low	1		
	Mezzo	Low	0		
	Help APORTA continue to make handwoven/knit...	Low	1		
	Music - Comedy - Album!	High	1		



#12

Query 16 with function:

select name, level\_amount(levels), status from main;

Output:

	name	goal_deflection(goal,pledged)	status
▶	WHILE THE TREES SLEEP	1045	1
	Educational Online Trading Card Game	-3980	0
	GETTING OVER - One son's search to finally kno...	535	1
	The Launch of FlyeGrlRoyalty &quot;The New N...	-3500	0
	Dinner Party - a short film about friendship... a...	82	1
	Mezzo	-720	0
	Help APORTA continue to make handwoven/knit...	180	1
	Music - Comedy - Album!	125	1

#13

Query 17 with function:

select name, goal\_deflection(goal,pledged), status from main;

Output:

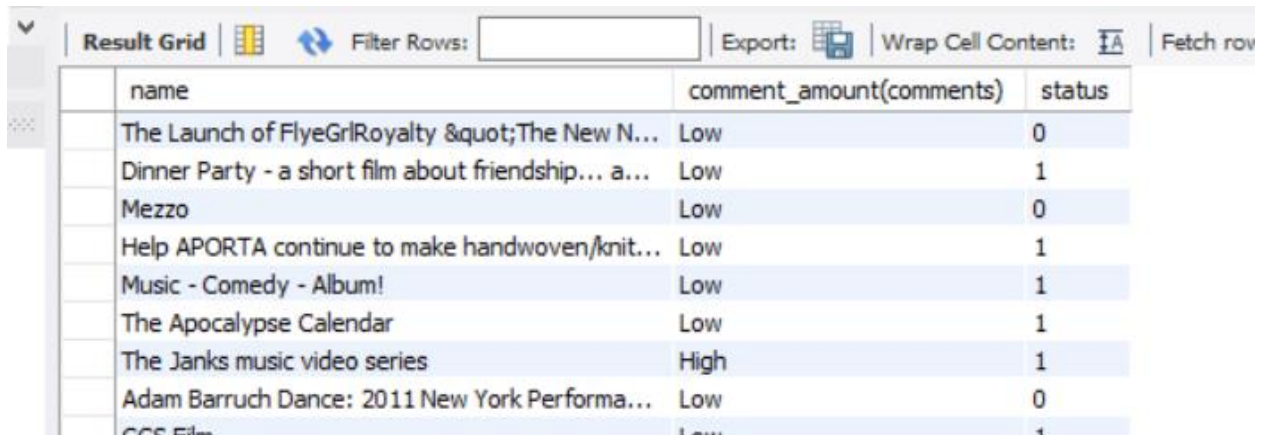
	name	comment_amount(comments)	status
▶	WHILE THE TREES SLEEP	Low	1
	Educational Online Trading Card Game	Low	0
	GETTING OVER - One son's search to finally kno...	Low	1
	The Launch of FlyeGrlRoyalty &quot;The New N...	Low	0
	Dinner Party - a short film about friendship... a...	Low	1
	Mezzo	Low	0
	Help APORTA continue to make handwoven/knit...	Low	1
	Music - Comedy - Album!	Low	1
	The Apocalypse Calendar	Low	1

#14

Query 18 with function:

select name, comment\_amount(comments), status from main;

Output:



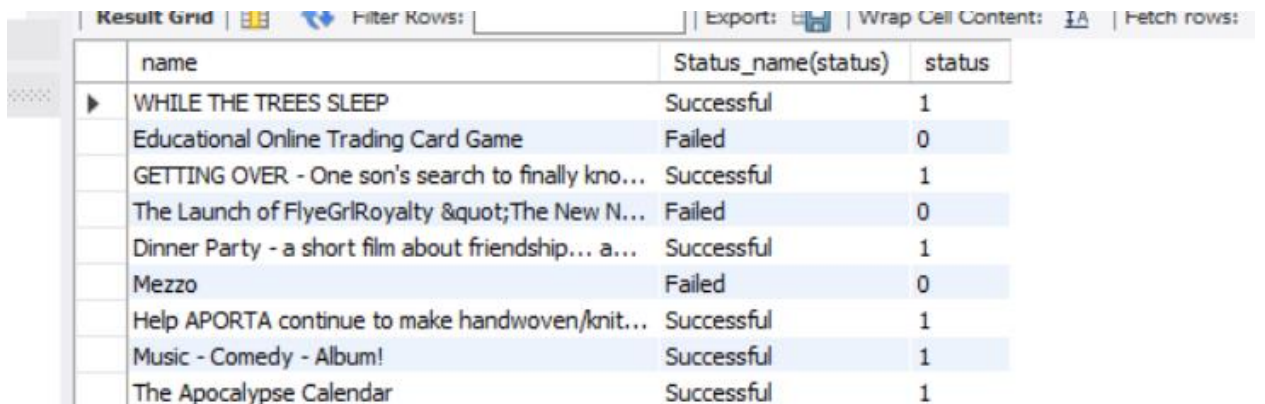
name	comment_amount(comments)	status
The Launch of FlyeGrlRoyalty &quot;The New N...	Low	0
Dinner Party - a short film about friendship... a...	Low	1
Mezzo	Low	0
Help APORTA continue to make handwoven/knit...	Low	1
Music - Comedy - Album!	Low	1
The Apocalypse Calendar	Low	1
The Janks music video series	High	1
Adam Barruch Dance: 2011 New York Performa...	Low	0
CCS Film	Low	1

#15

Query 19 with function:

select name, Status\_name(status), status from main;

Output:



name	Status_name(status)	status
WHILE THE TREES SLEEP	Successful	1
Educational Online Trading Card Game	Failed	0
GETTING OVER - One son's search to finally kno...	Successful	1
The Launch of FlyeGrlRoyalty &quot;The New N...	Failed	0
Dinner Party - a short film about friendship... a...	Successful	1
Mezzo	Failed	0
Help APORTA continue to make handwoven/knit...	Successful	1
Music - Comedy - Album!	Successful	1
The Apocalypse Calendar	Successful	1

# Functions in SQL

## Function 1:

Name:  The name of the routine is parsed automatically from the DDL statement. The DDL is parsed automatically while you type.

DDL:

```
1 • CREATE DEFINER='root'@'localhost' FUNCTION `backers_amount`(a1 int(11)) RETURNS text CHARSET utf8mb4
2     DETERMINISTIC
3     BEGIN
4     declare improvement text;
5
6     If a1>500 then
7         set improvement='High';
8     elseif(a1<=500) then
9         set improvement='Low';
10    end if;
11
12    RETURN(improvement);
13    END
```

## Function 2:

DDL:

```
1 • CREATE DEFINER='root'@'localhost' FUNCTION `comment_amount`(a1 int(11)) RETURNS text CHARSET utf8mb4
2     DETERMINISTIC
3     BEGIN
4     declare improvement text;
5
6     If a1>7 then
7         set improvement='High';
8     elseif(a1<=7) then
9         set improvement='Low';
10    end if;
11
12    RETURN(improvement);
13    END
```

## Function 3:

Name:  statement. The DDL is parsed automatically while you type.

DDL:

```
1 • CREATE DEFINER='root'@'localhost' FUNCTION `goal_deflection`(a1 int(11), a2 int(11)) RETURNS text CHARSET utf8m
2     DETERMINISTIC
3     BEGIN
4     declare improvement int(11);
5
6
7     RETURN(a2-a1);
8     END
```

## Function4:

```
1 • CREATE DEFINER='root'@'localhost' FUNCTION `goal_status`(a1 int(11)) RETURNS text CHARSET utf8mb4
2     DETERMINISTIC
3 BEGIN
4     declare improvement text;
5
6     If a1>5000 then
7         set improvement='High';
8     elseif(a1<=5000) then
9         set improvement='Low';
10    end if;
11
12    RETURN(improvement);
13 END
```

## Function 5:

DDL:

```
1 • CREATE DEFINER='root'@'localhost' FUNCTION `level_amount`(a1 int(11)) RETURNS text CHARSET utf8mb4
2     DETERMINISTIC
3 BEGIN
4     declare improvement text;
5
6     If a1>7 then
7         set improvement='High';
8     elseif(a1<=7) then
9         set improvement='Low';
10    end if;
11
12    RETURN(improvement);
13 END
```

## Function 6:

DDL:

```
1 • CREATE DEFINER='root'@'localhost' FUNCTION `Status_Name`(a1 int(11)) RETURNS text CHARSET utf8mb4
2     DETERMINISTIC
3 BEGIN
4     declare improvement text;
5
6     If a1=1 then
7         set improvement='Successful';
8     elseif(a1=0) then
9         set improvement='Failed';
10    end if;
11
12    RETURN(improvement);
13 END
```

# Views in SQL:

## View 1:

```
DDL:
1 • CREATE
2     ALGORITHM = UNDEFINED
3     DEFINER = `root`@`localhost`
4     SQL SECURITY DEFINER
5     VIEW `project`.`view1` AS
6     SELECT
7         `project`.`main`.`category` AS `Category`,
8         (AVG(`project`.`main`.`status`) * 100) AS `percentage`
9     FROM
10        `project`.`main`
11     GROUP BY `project`.`main`.`category`
```

## View 2:

```
1 • CREATE
2     ALGORITHM = UNDEFINED
3     DEFINER = `root`@`localhost`
4     SQL SECURITY DEFINER
5     VIEW `project`.`view2` AS
6     SELECT
7         `project`.`state`.`State` AS `state`,
8         SUM(`project`.`main`.`goal`) AS `sum(goal)`
9     FROM
10        (`project`.`main`
11     LEFT JOIN `project`.`state` ON ((`project`.`main`.`state` = `project`.`state`.`State_id`)))
12     WHERE
13         (`project`.`main`.`status` = 1)
14     GROUP BY `project`.`state`.`State`
15     ORDER BY SUM(`project`.`main`.`goal`) DESC
```

## View 3:

```
1 • CREATE
2     ALGORITHM = UNDEFINED
3     DEFINER = `root`@`localhost`
4     SQL SECURITY DEFINER
5     VIEW `project`.`view3` AS
6     SELECT
7         `project`.`subcategory`.`Subcategory_Name` AS `subcategory_name`,
8         SUM(`project`.`main`.`backers`) AS `Trending`
9     FROM
10        (`project`.`main`
11     LEFT JOIN `project`.`subcategory` ON ((`project`.`main`.`subcategory` = `project`.`subcategory`.`Subcategory_id`)))
12     GROUP BY `project`.`main`.`subcategory`
13     ORDER BY SUM(`project`.`main`.`backers`) DESC
```

## View 4:

```
1 • CREATE
2     ALGORITHM = UNDEFINED
3     DEFINER = `root`@`localhost`
4     SQL SECURITY DEFINER
5     VIEW `project`.`view4` AS
6     SELECT
7         `project`.`main`.`unique_id` AS `unique_id`,
8         `project`.`main`.`project id` AS `project id`,
9         `project`.`main`.`name` AS `name`,
10        `project`.`main`.`url` AS `url`,
11        `project`.`main`.`category` AS `category`,
12        `project`.`main`.`subcategory` AS `subcategory`,
13        `project`.`main`.`status` AS `status`,
14        `project`.`main`.`goal` AS `goal`,
15        `project`.`main`.`pledged` AS `pledged`,
16        `project`.`main`.`funded percentage` AS `funded percentage`,
17        `project`.`main`.`backers` AS `backers`,
```

## View 5:

```
1 • CREATE
2     ALGORITHM = UNDEFINED
3     DEFINER = `root`@`localhost`
4     SQL SECURITY DEFINER
5     VIEW `project`.`views5` AS
6     SELECT
7         AVG(`project`.`main`.`goal`) AS `Average_Goal_of_Successful_Project`
8     FROM
9         `project`.`main`
10    WHERE
11        (`project`.`main`.`status` = 1)
```

## View 6:

```
1 • CREATE
2     ALGORITHM = UNDEFINED
3     DEFINER = `root`@`localhost`
4     SQL SECURITY DEFINER
5     VIEW `project`.`view6` AS
6     SELECT
7         `project`.`main`.`unique_id` AS `unique_id`,
8         `project`.`main`.`project id` AS `project id`,
9         `project`.`main`.`name` AS `name`,
10        `project`.`main`.`url` AS `url`,
11        `project`.`main`.`category` AS `category`,
12        `project`.`main`.`subcategory` AS `subcategory`,
13        `project`.`main`.`status` AS `status`,
14        `project`.`main`.`goal` AS `goal`,
15        `project`.`main`.`pledged` AS `pledged`,
16        `project`.`main`.`funded percentage` AS `funded percentage`,
17        `project`.`main`.`backers` AS `backers`,
```

## References:

- **Kickstarter** (<https://www.kickstarter.com/>)
- **Predicting the success of Kickstarter campaigns** (<https://towardsdatascience.com/predicting-the-success-of-kickstarter-campaigns-3f4a976419b9>)
- **Kaggle** (<https://www.kaggle.com/parienza/kickstarter>)

## License

The MIT License Copyright 2019 Parth Rana, Shaishav Shah, Anshul balamwar

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Creative Commons The text in the document by Parth Rana, Shaishav Shah, Anshul balamwar. This work is licensed under the Creative Commons Attribution 3.0 United States License. To view a copy of this license, visit <http://creativecommons.org/licenses/by/3.0/us/> or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.