SQL Case Study-2

Deepak Singh
3 min read · Just now

Listen Share ••• More

Play-store-data-sql-case-study

Dataset- https://github.com/dk1coding1zone/SQL-Case-Study-2/blob/main/playstore.csv

Overview:

Category	Rating	Reviews	Size	Installs	Type	Price	Content_Rating	Genres	Last_Updated	Current_Ver	Android_Ver
ART_AND_DESIGN	4.1	159	19M	10000	Free	0	Everyone	Art & Design	07-01-2018	1.0.0	4.0.3 and up
ART_AND_DESIGN	3.9	967	14M	500000	Free	0	Everyone	Art & Design;Pretend Play	15-01-2018	2.0.0	4.0.3 and up

Task-1:

You're working as a market analyst for a mobile app development company. Your task is to identify the most promising categories(TOP 5) for launching new free apps based on their average ratings.

```
select category, round(avg(rating),2) as average  from playstoredata where type
group by category
order by average desc
limit 5;
```

if error in loading the data then do data infiling eg. unable to load complete data on MySql

```
LOAD DATA INFILE "C:/ProgramData/MySQL/playstore.csv"
INTO TABLE playstoredata
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\r\n'
IGNORE 1 ROWS;
```

Task-2:

As a business strategist for a mobile app company, your objective is to pinpoint the three categories that generate the most revenue from paid apps. This calculation is based on the product of the app price and its number of installations.

```
select category, round(sum(revenue),2) as rev from
(
select *,Installs*Price as revenue from playstoredata where  type='Paid'
)t  group by category
order by rev desc
limit 3;
```

Category	rev
FAMILY	185997086.70
LIFESTYLE	57583939.40
GAME	40986840.88

Task-3:

As a data analyst for a gaming company, you're tasked with calculating the percentage of games within each category. This information will help the company

understand the distribution of gaming apps across different categories.

```
select * , (cnt/(select count(*) from playstoredata))*100 as 'percentage' from
(
select category , count(category) as 'cnt' from playstoredata group by category
)m
```

	category	cnt	percentage
▶	ART_AND_DESIGN	61	0.6517
	AUTO_AND_VEHICLES	73	0.7799
	BEAUTY	42	0.4487
	BOOKS_AND_REFERENCE	178	1.9017
	BUSINESS	303	3.2372
	COMICS	58	0.6197
	COMMUNICATION	328	3.5043
	DATING	195	2.0833
	EDUCATION	155	1.6560
	ENTERTAINMENT	149	1.5919

Task-4:

As a data analyst at a mobile app-focused market research firm, you'll recommend whether the company should develop paid or free apps for each category based on the ratings of that category.

```
with freeapp as
(
 select category, round(avg(rating),2) as 'avg_rating_free' from playstoredata
 group by category
),
paidapp as
(
 select category, round(avg(rating),2) as 'avg_rating_paid' from playstoredata
 group by category
)

select *, if(avg_rating_free>avg_rating_paid,'Develop Free app','Develop Paid a
(
select f.category,f.avg_rating_free, p.avg_rating_paid  from freeapp as f inner
)k
```

category	avg_rating_free	avg_rating_paid	Development
ART_AND_DESIGN	4.36	4.73	Develop Paid app
AUTO_AND_VEHICLES	4.18	4.60	Develop Paid app
BOOKS_AND_REFERENCE	4.35	4.28	Develop Free app
BUSINESS	4.12	4.20	Develop Paid app
COMMUNICATION	4.17	4.06	Develop Free app
DATING	3.98	3.63	Develop Free app
EDUCATION	4.38	4.75	Develop Paid app
ENTERTAINMENT	4.12	4.60	Develop Paid app
FINANCE	4.14	3.83	Develop Free app
FOOD_AND_DRINK	4.16	4.35	Develop Paid app

Task-5:

Suppose you're a database administrator, your databases have been hacked and hackers are changing price of certain apps on the database , its taking long for IT team to neutralize the hack , however you as a responsible manager dont want your data to be changed , do some measure where the changes in price can be recorded as you cant stop hackers from making changes

```
-- creating table.
CREATE TABLE PriceChangeLog (
    App VARCHAR(255),
    Old_Price DECIMAL(10, 2),
    New_Price DECIMAL(10, 2),
    Operation_Type VARCHAR(10),
    Operation_Date TIMESTAMP
);

create table play as
SELECT * FROM PLAYSTORE

-- for update
DELIMITER //
CREATE TRIGGER price_change_update
AFTER UPDATE ON play
FOR EACH ROW
BEGIN
    INSERT INTO pricechangelog (app, old_price, new_price, operation_type, oper
    VALUES (NEW.app, OLD.price, NEW.price, 'update', CURRENT_TIMESTAMP);
END;
//
DELIMITER ;

SET SQL_SAFE_UPDATES = 0;
UPDATE play
SET price = 4
WHERE app = 'Infinite Painter';
```

```
UPDATE play
SET price = 5
WHERE app = 'Sketch - Draw & Paint';


select * from play where app='Sketch - Draw & Paint'
```

Edit profile

Written by Deepak Singh

1 Follower

Artificial Intelligence & Machine Learning Enthusiast , and emerging into the field of Data Science.

More from Deepak Singh