Retailer App Case Study

INTRODUTION:

- Our Plantix Partner app allows Retailers to order supplies online. We are continuously working on optimizing the app.
- We have 3 tables attached.
- The login_logs table contains information about users logging in.
- The sales orders table contains information about orders made
- The sales_orders_items contains the specifics of each order.
- We have been provided with data for July 2021 and July 2022.

OBJECTIVE:

- To optimise the usage of Retailer App.
- To provide suggestions for improving the business on basis of Findings.

Creating View **sales_orders_items_price** to get total price column

Query:

```
-- creating view with total price
CREATE VIEW sales_orders_items_price AS
    (SELECT *,round((order_quantity_accepted * rate),2) AS price FROM sales_orders_items);

SELECT * FROM sales_orders_items_price;
```

View Output:

order_item_id	fk_product_id	fk_order_id	ordered_quantity	order_quantity_accepted	rate	price
62609	7818	119874	1	0	4993	0
62610	7607	119877	1	0	6203.4	0
62611	9916	119880	2	2	2904.41	5808.82
62612	11118	119880	2	2	2205	4410
62613	12702	119880	2	2	2200.85	4401.7
62614	11110	119883	2	0	2723	0
62615	3234	119886	1	0	10700	0

1. Which KPIs would you use to measure the performance of our app?

KPI to measure performance of our app are below:

A. Retention.

Query 1: Total Users in Each year

```
-- total users in each year
SELECT year(login_time) year, COUNT(DISTINCT user_id) total_users FROM login_logs GROUP BY year;
```

Output:

year	total_users
2021	10867
2022	13022

Query 2: Total retained users in 2022 from 2021

```
-- retained users in 2022
SELECT COUNT(DISTINCT user_id) retained_users FROM login_logs WHERE year(login_time)='2022' AND user_id IN
               (SELECT DISTINCT user_id FROM login_logs WHERE year(login_time)='2021');
```

Output:

retained_users
3607

Query3: Retention Rate

```
-- Retention Rate
SELECT round((
(SELECT COUNT(DISTINCT user_id) retained_users FROM login_logs WHERE year(login_time)='2022' AND user_id IN
               (SELECT DISTINCT user_id FROM login_logs WHERE year(login_time)='2021') )/
(SELECT COUNT(DISTINCT user_id) total_users FROM login_logs WHERE year(login_time)='2021')),2) as retention_rate;
```

Output:

retention_rate
0.33

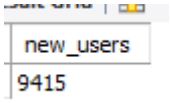
- We have 10867 users in 2021.
- Out off 13022 users in 2022, 3607 users are retained from last year.
- Our retentions rate is 33%

B. Churning

Query 1: Total New users in 2022

```
-- total new uesr in 2022
SELECT COUNT(DISTINCT user_id) new_users FROM login_logs WHERE year(login_time)='2022' AND user_id NOT IN
                (SELECT DISTINCT user_id FROM login_logs WHERE year(login_time)='2021');
```

Output:

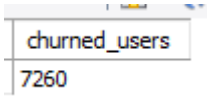


new_users
9415

Query 2: Total Churned Users

```
-- churned users of 2021
SELECT(
(SELECT COUNT(DISTINCT user_id) total_users FROM login_logs WHERE year(login_time)='2021')-
(SELECT COUNT(DISTINCT user_id) retained_users FROM login_logs WHERE year(login_time)='2022' AND user_id IN
                (SELECT DISTINCT user_id FROM login_logs WHERE year(login_time)='2021'))) AS churned_users;
```

Output:

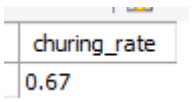


churned_users
7260

Query 3: Churning Rate

```
-- churning rate
WITH cte AS(
    SELECT round((
    (SELECT COUNT(DISTINCT user_id) retained_users FROM login_logs WHERE year(login_time)='2022' AND user_id IN
                    (SELECT DISTINCT user_id FROM login_logs WHERE year(login_time)='2021') )/
    (SELECT COUNT(DISTINCT user_id) total_users FROM login_logs WHERE year(login_time)='2021')),2) as retention_rate)
SELECT 1 - (SELECT retention_rate FROM cte) as churing_rate;
```

Output:



churing_rate
0.67

- Total New Users in 2022 are 9415
- Total 7265 users from 2021 does not continue in 2022
- Our Churning rate is 67%

C. Daily Active Users

Query:

```
-- C. Daily Active Users
SELECT year(login_time) year, day(login_time) day, COUNT(*) total_users FROM login_logs
GROUP BY year,day;
```

Output:

year	day	total_users
2021	1	7448
2021	2	8226
2021	3	7285
2021	4	4346
2021	5	9278
2021	6	11161
2021	7	10794
2021	8	10957
2021	9	10159
2021	10	10140
2021	11	5448
2021	12	11038
2021	13	10353
2021	14	10339
2021	15	10600

year	day	total_users
2021	16	9558
2021	17	8853
2021	18	4593
2021	19	10034
2021	20	9686
2021	21	10251
2021	22	8929
2021	23	8879
2021	24	8522
2021	25	4380
2021	26	8835
2021	27	9892
2021	28	9708
2021	29	10971
2021	30	10577

year	day	total_users
2022	1	9843
2022	2	11165
2022	3	6921
2022	4	13630
2022	5	13966
2022	6	11704
2022	7	17322
2022	8	14444
2022	9	12827
2022	10	6378
2022	11	13326
2022	12	14605
2022	13	17570
2022	14	15062
2022	15	16508

year	day	total_users
2022	16	13906
2022	17	9454
2022	18	15815
2022	19	15990
2022	20	17078
2022	21	15569
2022	22	16045
2022	23	13450
2022	24	6645
2022	25	14360
2022	26	11165
2022	27	12740
2022	28	12401
2022	29	13010
2022	30	12218

- Our count of total daily active users is changing frequently.

D. Daily No of users who ordered

Query:

```
-- D. Daily no of users who ordered
SELECT year(creation_time) year, day(creation_time) day, COUNT(*) total_users FROM sales_orders
GROUP BY year,day;
```

Output:

year	day	total_users
2021	1	150
2021	2	186
2021	3	163
2021	4	83
2021	5	295
2021	6	249
2021	7	250
2021	8	232
2021	9	236
2021	10	230
2021	11	105
2021	12	315
2021	13	215
2021	14	209
2021	15	254

year	day	total_users
2021	16	180
2021	17	198
2021	18	58
2021	19	260
2021	20	255
2021	21	254
2021	22	208
2021	23	223
2021	24	207
2021	25	65
2021	26	199
2021	27	236
2021	28	240
2021	29	209
2021	30	252

year	day	total_users
2022	1	181
2022	2	240
2022	3	121
2022	4	348
2022	5	308
2022	6	220
2022	7	278
2022	8	263
2022	9	206
2022	10	70
2022	11	294
2022	12	311
2022	13	286
2022	14	277
2022	15	314

year	day	total_users
2022	16	320
2022	17	124
2022	18	327
2022	19	294
2022	20	302
2022	21	279
2022	22	327
2022	23	231
2022	24	80
2022	25	300
2022	26	195
2022	27	231
2022	28	251
2022	29	217
2022	30	219

- Our total no of users who ordered is changing daily

E. Top 10 buying users (Lifetime Value)

Query:

```
-- E. Top 10 buying users (Lifetime Value)
SELECT fk_buyer_id AS user_id,
        round(SUM(price),2) total_sales
FROM sales_orders_items_price sip
JOIN sales_orders so ON sip.fk_order_id = so.order_id
GROUP BY fk_buyer_id
ORDER BY total_sales DESC
LIMIT 10;
```

Output:

user_id	total_sales
35700	4575926.29
97984	4369117.54
32444	3778763
40988	3254441.91
2124	3148806
160840	3095834
225336	2865000
182356	2344478
28656	2288931.4
25932	1882296

- User with user id 35700 has given us the top sales till now.

F. Top 10 visited users (Lifetime)

Query:

```
-- F. Top 10 visited users (Lifetime)
SELECT user_id, COUNT(*) total_visits FROM login_logs
GROUP BY user_id
ORDER BY total_visits DESC
LIMIT 10;
```

Output:

user_id	total_visits
63652	3868
209276	2471
51192	2097
43108	1899
42292	1714
97272	1711
220464	1354
46516	1300
215084	1293
43180	1286

- User with user id 63652 has visited most of the times on our App.

2. Prepare a report regarding our growth between the 2 years. Please try to answer the following questions:

A. Did our business grow?

Query:

```
-- A. Did our business grow?
WITH cte AS (
SELECT year(creation_time) year, round(SUM(price),2) total_revenue FROM sales_orders AS so
JOIN sales_orders_items_price AS sip  ON so.order_id = sip.fk_order_id
GROUP BY year)
SELECT * , total_revenue - LAG(total_revenue,1) OVER() revenue_growth,
round((total_revenue/LAG(total_revenue) OVER() *100)-100,2) 'growth%' FROM cte;
```

Output:

year	total_revenue	revenue_growth	growth%
2021	60315971.07	NULL	NULL
2022	116858178.44	56542207.37	93.74

- Total revenue for year 2021 is *60315971.07*
- Total revenue for year 2022 is *116858178.44*
- Total revenue growth is *56542207.37* with 94% growth.
- So, we can say that our business has grown.

B. Does our app perform better now?

Query 1: Sales orders performance optimization

```
-- Sales orders performance optimization
SELECT x.sales_order_status,
x.total_orders AS total_orders_2022,
y.total_orders AS total_orders_2021,
round(((x.total_orders/y.total_orders)*100)-100,2) AS 'growth_%_21_to_22'
FROM
        (SELECT year(creation_time) year, sales_order_status, COUNT(*) AS total_orders FROM sales_orders
        WHERE year(creation_time) = '2022'
        GROUP BY year, sales_order_status) x
LEFT JOIN
        (SELECT year(creation_time) year, sales_order_status, COUNT(*) AS total_orders FROM sales_orders
        WHERE year(creation_time) = '2021'
        GROUP BY year, sales_order_status) y
ON x.sales_order_status = y.sales_order_status;
```

Output:

sales_order_status	total_orders_2022	total_orders_2021	growth_%_21_to_22
Rejected	3899	3764	3.59
Shipped	3489	2434	43.34
Pending	25	18	38.89
Review	1	NULL	NULL

- Total rejected orders in 2022 are 3899 and in 2021 are 3764, we were able keep rejected orders low at 4%.
- Total Shipped orders in 2022 are 3489 and in 2021 are 2434, which are 44% more than previous year.
- Total Pending orders in 2022 are 25 and in 2021 are 18, which we were able keep low.
- Only one order was reviewed in 2022.

Query 2: How many order quantity accepted out off total ordered quantity

```
-- How many order qunatity accepted out off total ordered quantity
WITH cte AS
    (SELECT year(creation_time) year,
    SUM(ordered_quantity) total_ordered_items ,
    SUM(order_quantity_accepted) total_accepted_items
    FROM sales_orders_items soi
    JOIN sales_orders so ON so.order_id = soi.fk_order_id
    GROUP BY year)
SELECT year,
total_ordered_items as ttl_ord_itms,
round((total_ordered_items/LAG(total_ordered_items) OVER()*100)-100,2) AS 'ord_itms_growth%',
total_accepted_items as ttl_acc_itms,
round((total_accepted_items/LAG(total_accepted_items) OVER()*100)-100,2) AS 'ord_acc_growth%',
round(((total_ordered_items - total_accepted_items)/
            LAG(total_ordered_items - total_accepted_items) OVER() *100)-100,2)
            AS total_rejected_items
FROM cte;
```

Output:

year	ttl_ord_itms	ord_itms_growth%	ttl_acc_itms	ord_acc_growth%	total_rejected_items
2021	31738	NULL	11485	NULL	NULL
2022	34900	9.96	18301	59.35	-18.04

- Total orders Items growth is 10%
- Total Accepted orders growth is 60%
- Total Rejected orders reduced by 18%.

C. Did our user base grow?

Query:

```
-- c. Did our user base grow?
SELECT x.year,
total_users,
round((total_users/LAG(total_users)OVER() *100)-100,2)  AS total_users_gain,
ordered_users,
round((ordered_users/LAG(ordered_users)OVER() *100)-100,2) AS ordered_users_gain
FROM
    (SELECT year(login_time) year, COUNT(*) total_users FROM login_logs
    GROUP BY year) AS x,
    (SELECT year(creation_time) year, COUNT(*) ordered_users FROM sales_orders
    GROUP BY year) y
WHERE x.year = y.year;
```

Output:

year	total_users	total_users_gain	ordered_users	ordered_users_gain
2021	271240	NULL	6216	NULL
2022	395117	45.67	7414	19.27

- Our total user base grown by 46%
- Total users who ordered grown by 20%
- So we can say that our user base has grown.

3. What are our top-selling products in each of the two years? Can you draw some insight from this?

Query 1: Top 10 sold products

```
-- Top 10 sold products
WITH cte AS(
    SELECT year(creation_time) year, fk_product_id AS product_id, COUNT(*) AS total_sold
    FROM sales_orders_items_price sip
    JOIN sales_orders so ON so.order_id = fk_order_id
    GROUP BY fk_product_id,year
    ORDER BY total_sold DESC)
SELECT year, product_id, total_sold FROM(
                    SELECT *, ROW_NUMBER() OVER(PARTITION BY year) AS rn FROM cte) x
WHERE rn <= 10;
```

Output:

year	product_id	total_sold
2021	10235	550
2021	8444	337
2021	1548	300
2021	1041	268
2021	9925	249
2021	8425	245
2021	10975	212
2021	8219	204
2021	8210	187
2021	11118	186

2022	8444	485
2022	10235	358
2022	1041	356
2022	8425	345
2022	3610	335
2022	10975	257
2022	10224	236
2022	1038	225
2022	12701	224
2022	3615	205

- Top times Selling product of 2021 is 10235, followed by 8444
- Top times Selling product of 2022 is 8444, followed by 10235.

Query 2: Top 10 products which gave highest business

```
-- Top 10 products which gave highest business
WITH cte AS(
    SELECT year(creation_time) year, fk_product_id AS product_id, round(SUM(price),2) AS total_sale
    FROM sales_orders_items_price sip
    JOIN sales_orders so ON so.order_id = fk_order_id
    GROUP BY fk_product_id, year
    ORDER BY total_sale DESC)
SELECT year, product_id, total_sale FROM(
                    SELECT *, ROW_NUMBER() OVER(PARTITION BY year) AS rn FROM cte) x
WHERE rn <= 10;
```

Output:

year	product_id	total_sale
2021	8219	11337874.39
2021	8210	5328288.5
2021	7640	2696077.6
2021	1548	2563695.52
2021	10975	2353035.3
2021	9925	1794376.67
2021	3610	1720389
2021	10235	1415069.45
2021	8444	1388173.4
2021	1038	1275489.12

2022	12547	9716031
2022	3610	9520258
2022	8219	9163587
2022	8444	6982455
2022	7640	6508070
2022	12652	5405502
2022	8221	3632474.01
2022	8210	3317442
2022	12749	3238739
2022	10975	3238187

- Top business giving product if 2021 is 8219, followed by 8210
- Top business giving product if 2022 is 12547, followed by 3610

4. Looking at July 2021 data, what do you think is our biggest problem and how would you recommend fixing it?

Query :

```
-- 4. Looking at July 2021 data, what do you think is our biggest problem and how would you
SELECT total_ordered_items,
total_accepted_items,
round(total_accepted_items/total_ordered_items*100,2) AS 'Acc_%',
total_ordered_items - total_accepted_items AS total_rejected_items,
round(((total_ordered_items - total_accepted_items)/total_ordered_items)*100,2) AS 'rej_%'
FROM
    (SELECT year(creation_time) year,
    SUM(ordered_quantity) total_ordered_items ,
    SUM(order_quantity_accepted) total_accepted_items
    FROM sales_orders_items soi
    JOIN sales_orders so ON so.order_id = soi.fk_order_id
    GROUP BY year)x;
```

Output:

total_ordered_items	total_accepted_items	Acc_%	total_rejected_items	rej_%
31738	11485	36.19	20253	63.81
34900	18301	52.44	16599	47.56

- We can see that rejected items are like 63% of total orders, which was our biggest problem in 2021
- We are trying improve it 2022, we have accepted 52% of total orders.

5. Does the login frequency affect the number of orders made?

Query 1: Out off total users

```
-- A. Out off total users
WITH cte AS (
    SELECT x.year, x.day, total_users, ordered_users FROM
    (SELECT year(login_time) year, day(login_time) day, COUNT(*) total_users FROM login_logs
    GROUP BY year, day) AS x,
    (SELECT year(creation_time) year, day(creation_time) day, COUNT(*) ordered_users FROM sales_orders
    GROUP BY year, day) y
    WHERE x.day = y.day AND x.year = y.year )
SELECT *, round((ordered_users/total_users)*100,2) AS '%total_ordred_users' FROM cte;
```

Output:

year	day	total_users	ordered_users	%total_ordred_users
2021	1	7448	150	2.01
2021	2	8226	186	2.26
2021	3	7285	163	2.24
2021	4	4346	83	1.91
2021	5	9278	295	3.18
2021	6	11161	249	2.23
2021	7	10794	250	2.32
2021	8	10957	232	2.12
2021	9	10159	236	2.32
2021	10	10140	230	2.27
2021	11	5448	105	1.93
2021	12	11038	315	2.85
2021	13	10353	215	2.08
2021	14	10339	209	2.02

- Out off total user who have visited on our App, more or less 2~4 % users are ordering.

Query 2: Out of total users who ordered

```
-- B. Out of total users who ordered
WITH cte AS (
    SELECT x.year, x.day, total_users, ordered_users FROM
    (SELECT year(login_time) year, day(login_time) day, COUNT(*) total_users FROM login_logs
    WHERE user_id IN ( SELECT fk_buyer_id FROM sales_orders)
    GROUP BY year, day) AS x,
    (SELECT year(creation_time) year, day(creation_time) day, COUNT(*) ordered_users FROM sales_orders
    GROUP BY year, day) y
    WHERE x.day = y.day AND x.year = y.year )
SELECT *, round((ordered_users/total_users)*100,2) AS '%total_ordred_users' FROM cte;
```

Output:

year	day	total_users	ordered_users	%total_ordred_users
2021	1	4176	150	3.59
2021	2	4998	186	3.72
2021	3	4252	163	3.83
2021	4	2647	83	3.14
2021	5	6194	295	4.76
2021	6	7240	249	3.44
2021	7	7031	250	3.56
2021	8	7405	232	3.13
2021	9	6860	236	3.44
2021	10	6648	230	3.46
2021	11	3393	105	3.09
2021	12	7624	315	4.13
2021	13	7234	215	2.97
2021	14	6923	209	3.02

sult 197

- Out off total users who have ordered from our App more or less 3~6% users are ordering.

Conclusion:

We have worked on optimisation of the usage of our partner Retailer App and providing suggestions for improving the business on basis of Findings:

- We have 13022 users in 2022 and 10867 users in 2021 with retentions rate of 33% and Churning rate of 67%. So, we need to increase retention rate.
- Our daily active users and daily users who orders changing frequently good thing is it increasing trend, We need steadilly increase it.
- Our business has grown by 94% and user base growth by 46% with better performance of our app, we were able decrease the no of rejected orders.
- 8444, 10235, 1041, 8425, 10975 are top selling products.
- Our biggest problem in 2021 was number rejected items and we are improving it, which we have achieved in 2022.
- Login frequency is not that much affecting the number of orders made.