Does failing increase risk of churn?

We would get the answer by finding following factors -

- Calculate no. of players failing and immediately quitting
- Calculate no. of players failing on last session they played
- Calculate no. of players failing on last stage they played
- Calculate no. of players failing on last level they played

Metrics calculated -

- 1.DAU (daily active users)
- 2.Maximum days most players left
- 3.Maximum level players are playing
- 4.No. of fails on particular level and stage
- 5.No. of fails on particular level
- 6.Failing ratio (total fails per total starts)

-Fails by last event_datetime

7.No. of players failed after which they never played the game (players who failed and immediately quit the game)

-Fails by last session_id

8.No. of players failed in a session after which they never played the game (at least 1 fail in last session)

-Fails by last level

9.No. of players failed in their last level after which they never played the game (at least 1 fail in their last level played)

-Fails by last stage

10.No. of players failed in a stage after which they never played the game (at least 1 fail in last stage)

- 11.Levels on which most players failed and never played the game (by last event_datetime)
- 12.Frequency of count of fails on their last level (so,most people fail 1 time in their last level played)
- 13. Exploring Players dataset

```
CREATE DATABASE kwalee;
USE kwalee;

CREATE TABLE levels (
event_datetime DATETIME,
player_id VARCHAR(35),
levelno INT,
stageno INT,
status_ VARCHAR(10),
session_id VARCHAR(35) );
```

Extract csv file into table

```
LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/level_progress.csv"
INTO TABLE levels
COLUMNS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 LINES
( event_datetime,player_id,levelno,stageno,status_,session_id );



SELECT * FROM levels;
```

1. DAU (daily active users)

```
# SELECT AVG(c) FROM (

SELECT DATE(event_datetime)AS d ,DAYNAME(event_datetime),COUNT(player_id) AS c
FROM levels
GROUP BY DATE(event_datetime),DAYNAME(event_datetime)
ORDER BY DATE(event_datetime) ;

 #)AS table1;
```

2. Maximum days most players played the game

```
SELECT days_played ,COUNT(days_played) FROM
(
SELECT player_id, COUNT(DISTINCT DAY(event_datetime)) AS days_played ,
COUNT(session_id) ,MAX(levelno) AS maxlevel
FROM levels
GROUP BY player_id
ORDER BY 2,3 DESC,4 DESC
) AS table1
GROUP BY days_played;
```

days_played	count(days_played)
1	2955
2	1146
3	433
4	203
5	87
6	55

3. Maximum levels players played

```
SELECT max_level, COUNT(max_level) AS player_count FROM
(
SELECT player_id, MAX(levelno) AS max_level FROM levels
GROUP BY player_id
ORDER BY 2 DESC
)AS table1
GROUP BY max_level
ORDER BY 1,2 DESC;
```

max_level	player_count
1	1231
2	1514
3	927
4	504
5	306

4.No. of fails on particular level and stage

```
SELECT levelno, stageno,COUNT(status_)
FROM levels
WHERE status_='fail'
GROUP BY levelno, stageno
ORDER BY 1,2;
```

5.No. of Fails on level

```
SELECT levelno,
COUNT(status_)
FROM levels
WHERE status_='fail'
GROUP BY levelno
ORDER BY 2 DESC;
```

levelno	count(status_)
4	1623
1	1256
2	1243
5	1144
3	838
6	498
7	279
8	255

6. Failing ratio (total fails per total starts)

```
WITH
table1 AS (
SELECT levelno, stageno, COUNT(status_) AS fail FROM levels
WHERE status_='fail'
GROUP BY levelno, stageno, status_
ORDER BY 1 ),

table2 AS (
SELECT levelno, stageno, COUNT(status_) AS total FROM levels
WHERE status_='start'
GROUP BY levelno, stageno, status_
ORDER BY 1 )

SELECT t1.levelno as Level_number, t1.stageno as Stage_number, t1.fail as Total_fails
,t2.total as Total_starts,CAST( (t1.fail/t2.total)*100  AS DECIMAL(10,2) ) AS Fails_ratio
FROM table1 AS t1
JOIN table2 t2
ON t1.levelno=t2.levelno AND t1.stageno=t2.stageno
ORDER BY 1,2;
```

Failing and Quitting

7.No. of players failed after which they never played the game (by last event_datetime) (players who failed and immediately quit the game)

```
WITH table1 AS
(
SELECT player_id,  status_, levelno, stageno, session_id, event_datetime,
ROW_NUMBER()  OVER ( PARTITION BY player_id ORDER BY event_datetime DESC) AS rnk
FROM levels
)
 SELECT status_ AS players_last_event_time,COUNT(status_) AS no_of_players FROM table1
WHERE rnk BETWEEN 0 AND 1
GROUP BY status_;
```

players_last_event_time	no_of_players
start	3123
fail	838
complete	978

8.No. of players failed in their last session after which they never played the game (at least 1 fail in last session)

```
WITH table1 AS
(
SELECT player_id, levelno, stageno, status_, session_id, event_datetime,
DENSE_RANK()  OVER ( PARTITION BY player_id ORDER BY player_id,session_id) rnk
FROM levels
)
SELECT DISTINCT player_id
FROM table1
WHERE status_='fail' AND rnk BETWEEN 0 AND 1;
```

Players count - 963

9.No. of players failed in their last level after which they never played the game (at least 1 fail in their last level played)

```
WITH table1 AS (
SELECT player_id, levelno, stageno, status_, session_id, event_datetime,
DENSE_RANK()  OVER( PARTITION BY player_id ORDER BY  player_id,levelno DESC) rnk
FROM levels
)
#select count(*) from table1
SELECT DISTINCT player_id FROM table1
WHERE status_='fail' AND rnk=1;
```

Players count - 1743

10.No. of players failed in a stage after which they never played the game (at least 1 fail in last stage)

```
WITH table1 AS (
SELECT player_id, levelno, stageno, status_,
DENSE_RANK()  OVER ( PARTITION BY player_id ORDER BY player_id, levelno DESC,stageno
DESC) rnk
FROM levels
)
SELECT DISTINCT player_id
FROM table1
WHERE status_='fail' AND rnk=1;
```

Players count - 1350

11.Count of level/stage on which players failed and never played the game (by last event_time)

```
SELECT levelno, count(levelno) AS players_failing FROM (
WITH table1 AS
(
SELECT player_id, status_, levelno, stageno, event_datetime,
ROW_NUMBER()  OVER ( PARTITION BY player_id ORDER BY event_datetime DESC) AS rnk
FROM levels
)
SELECT player_id, status_, levelno, stageno, event_datetime,rnk
FROM table1
WHERE status_='fail' AND rnk=1
)AS table2
GROUP BY levelno
ORDER BY 2 DESC;
```

levelno	players_failing
4	168
2	160
1	140
3	126
5	111
6	47
7	21

12.Frequency of count of fails on their last level (so, people mostly fail 1 time in their last level played)

fails_on_last_level	frequency
1	1208
2	328
3	120
4	52
5	13

```
select fails_on_last_level, count(fails_on_last_level) as frequency_of_fails from (

WITH table1 AS (
select player_id,levelno,stageno,status_,
DENSE_RANK() OVER(PARTITION BY player_id ORDER BY player_id,levelno DESC) rnk
FROM levels
)
SELECT player_id ,levelno AS last_level,COUNT(status_)AS fails_on_last_level
FROM table1
WHERE status_='fail' AND rnk=1
GROUP BY player_id,levelno

)AS table3
GROUP BY fails_on_last_level
ORDER BY 2 DESC;
```

13.Exploring Players dataset

```
create database kwalee;
use kwalee;

create table players (
install_datetime datetime,
player_id varchar(35),
platform varchar(10),
country varchar(5),
screen_size double,
system_memory int
);

desc players;

LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/players.csv"
INTO TABLE players
COLUMNS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 LINES
(install_datetime,player_id,platform,country,screen_size,system_memory);
```

 # EXPLORING DATA

```
select * from players;


select distinct country from players;
select country,count(player_id) from players group by country order by 2 desc;


select platform,count(platform) from players group by platform;
```

```
select system_memory,count(system_memory) as count_memory
from players
where system_memory between 2000 and 4000
group by system_memory order by 2 desc;


select sum(count_memory) from
(
select system_memory,count(system_memory) as count_memory
from players
where system_memory between 2600 and 4000
group by system_memory order by 2 desc
)
as table1;

select screen_size,count(screen_size) from players group by screen_size order by 2 desc;

select date(install_datetime) from players order by install_datetime ;
```

New daily installs

```
select date(install_datetime), count(install_datetime) as daily_installs
from players
group by DATE(install_datetime)
order by date(install_datetime);
```

Average new daily installs

```
with table2 as
(
select date(install_datetime), count(install_datetime) as daily_installs
from players
group by DATE(install_datetime)
order by date(install_datetime)
)  select avg(daily_installs) from table2;
```

Total players did not opened the app after installing (60 players)

```
select p.player_id ,l.event_datetime
from players p
left join levels l
on l.player_id=p.player_id
where event_datetime is NULL;
```

Failing and quitting relationship with platform

```
select platform, count(platform) from
(
with table1 as (
select player_id,levelno,stageno,status_,
dense_rank() over(partition by player_id order by player_id,levelno desc,stageno desc) rnk
from levels
                                )
select distinct t.player_id , p.platform as platform, system_memory
from table1 t
join players p
on t.player_id=p.player_id
where status_='fail' and rnk=1
)as table2
group by platform;
```

Failing and quitting relationship with system_memory

```
select system_memory,count(system_memory) from #platform, count(platform) from
(
with table1 as (
select player_id,levelno,stageno,status_,
dense_rank() over(partition by player_id order by player_id,levelno desc,session_id) rnk
from levels
                                )
select distinct t.player_id , p.platform as platform, system_memory
from table1 t
join players p
on t.player_id=p.player_id
where status_='fail' and rnk=1
)as table2
group by system_memory
order by 2 desc;
#group by platform;
```