Source Code

Q.1 Extract P_ID, Dev_ID, PName and Difficulty_level of all players at level 0

Solution SELECT pd.p_ID, Id.Dev_Id, pd.PName, Id.Difficulty from (pd inner join Id on pd.P Id.P_ID) where 'level' =0;

Q.2 Find Level1_code wise Avg_kill_Count where lives_earned is 2 and atleast 3 stages are crossed

Solution: SELECT pd.L1_code, avg (Id.kill_count), Id.lives_earned, Id.stages_crossed FROM pd JOIN Id on pd.P_ID = Id.P_ID where (lives_earned=2 and stages_crossed > =3);

Q.3 Find the Total Number Of Stages Crossed at each difficulty level where for Level2 with players use zm_series devices. Arrange the result in decsreasing order of total number of stage crossed.

Solution: Select sum (stages_crossed), difficulty, 'level', dev_ID from Id where (dev_ID like "zm%" and 'level' = 2) group by difficulty order by sum (stages_crossed) desc;

Q.4) Extract P_ID and the total number of unique dates forthose players who have played games on multiple days.

Solution: select P_ID, count(distinct start_time) AS unique_dates from Id group by P_ID having count(distinct start_time) >1;

Q.5) Find P_ID and Level wise sum of kill_counts where kill_count is grater than avg kill count for medium Difficulty

Solution: select P_ID, 'level', sum (kill_count) from Id where kill_count > (select avg(kill_count) from where difficulty ="medium") group by P_ID, 'level';

Q.6) Find Level and its corresponding 'Level_code wise sum of lives earned, excluding Level 0. Arrange in ascending order of level.

Solution: select Id.level, pd.L1_code, sum (Id.lives_earned) as total_lives_earned from Id inner join pd on Id.P_ID = pd.p_ID where Id.level> 0 group by Id.level, pd.L1_code order by Id.level asc;

Q.7) Find the top 3 scores based on each Dev_ID and rank them in increasing order using Row_Number. Display the difficulty as well.

Solution: select score, dev_ID, difficulty, row_number() over (partition by dev_ID order by score desc) as ranked from Id limit 3;

Q9) Find the top 5 scores based on each difficulty level and rank them in increasing order using Rank. Display Dev_ID as well.

Solution: select dev_ID, score, difficulty, rank() over (partition by difficulty order by score desc as ranked from id limit 5;

10. Find the device ID that is first logged in (based on start_datetime') for each player ('P_ID'). Output should contain player ID, device ID, and first login datetime.

Solution: select P_ID, dev_ID, min (start_time) from id group by dev_id, P_ID; select*from id;

11. For each player and date, determine how many kill_counts were played by the player so far. a) Using window functions b) Without window functions

select P_ID, date(start_time) as date, sum (kill_count) over (partition by P_ID order by start_time) as total_kill_countfrom ld;

12. Find the cumulative sum of stages crossed over 'start_datetime' for each 'P_ID`, excluding the most recent start_datetime

solution: select P_ID, start_time, stages_crossed, sum (stages_crossed) over (partition by P_ID order by start_time rows between unbounded preceding and 1 preceding) from Id;

13. Extract the top 3 highest sums of scores for each Dev_ID and the corresponding P_ID.

solution: select P_ID, dev_ID, sum (score), row_number() over (partition by dev_ID order by sum (score) desc) as ranked from Id group by dev_ID, P_ID limit 3;

14. Find players who scored more than 50% of the average score, scored by the sum of scores for each 'P_ID'.

solution: select P_ID, sum (score) from Id group by P_ID having sum (score) > 0.5* (select avg (score) from Id);

15. Create a stored procedure to find the top 'n headshots_count based on each Dev_ID and rank them in increasing order using Row_Number. Display the difficulty as well.

Solution: DELIMITER //CREATE PROCEDURE TOPN (IN n INT) BEGINselect dev_ID, headshots_count, difficulty, row_number() over (partition by dev_ID Order by headshots_count) as rankedfrom ID) as taskwhere ranked <=n; END//call TOPN(6)