

Department of Electrical and Computer Engineering
North South University



Database Design Project

Shooting Game Database

Submitted To:

Kamruddin Md. Nur (KMN1)

Department of ECE

Submitted by Group-9:

Md. Kayes Hossan Fuad - 1731040642

Tanzina Tazreen Meem - 1620199042

Abstract

Shooting games are a subgenre of action video games, which often test the player's spatial awareness, reflexes, and speed in both isolated single player or networked multiplayer environments. Shooter games encompass many subgenres that have the commonality of focusing on the actions of the avatar engaging in combat with a weapon against both code-driven NPC enemies or other avatars controlled by other players.

Usually, this weapon is a firearm or some other long-range weapon and can be used in combination with other tools such as grenades for indirect offense, armor for additional defense, or accessories such as telescopic sights to modify the behavior of the weapons. A common resource found in many shooter games is ammunition, armor or health, or upgrades that augment the player character's weapons.

Most commonly, the purpose of a shooter game is to shoot opponents and proceed through missions without the player's character being killed or dying as a result of the player's actions. A shooting game is a genre of video game where the focus is almost entirely on the defeat of the character's enemies using the weapons given to the player. More than 500+ million people play shooting games on pc, mobile, ps4, and other consoles. So Due to a lot of player management of information, it is so tough. A database of player information can make it easy to save all player info and stats.

Table of Contents

Chapter 1: Introduction.....	4
Chapter 2: Data Model	5-9
Chapter 3: Testing.....	10-14
Chapter 6: Discussion.....	15
Chapter 7: Conclusion.....	16
Chapter 8: Acknowledgement.....	17
Chapter 9: References.....	18

1. Introduction

To reiterate the whole idea why we require databases, in other words, the type of database we should use, to say the least. On shooting games, there can be a lot of player data and gaming stats and more info. There are a lot of video games in the world. The gaming Esports community is also available in many countries. Gaming has a bright future too. Game developer companies earn huge revenue from shooting games. But collecting player data and gaming resources need a strong database and now time to play online games for updating games and adding new features it can be possible from changing databases without changing gaming resources. Player stats, gaming mood, and reports and feedback can be stored on the database. We are planning a database which can be made simple to manage all player stats easily.

2. Data Model

We had done Database design before building the actual database to meet the needs of end-users within a given information-system that the database is intended to support. The database design defines the needed data and data structures that such a database comprises

The database is physically implemented using MySQL.

The database consists of seven tables. User_table, log_in, character_stat, multiplayer, messages, summary, battle_royale_

1. **User_table:** contains all the information of the players who played this game.
2. **log_in:** contains the login information of the players who log in to play the game.
3. **character_stat:** contain the information about avatars or legends that the players use during the game.
4. **Multiplayer:** contain information about gaming mode “multiplayer”.
5. Messages: contains the messages passed between players.
6. Summary: holds the summary about matches of “multiplayer” gaming mode.
7. battle_royale: contain information about gaming mode “battle_royale”.

Each entity(table) with its attributes can be described as follows along:

Database tables structures:

Table name: User_table

Field	Type	Null	Key	Default	Extra
User_Id	decimal(10,0)	NO	PRI	NULL	
User_Fname	varchar(50)	YES		NULL	
User_Lname	varchar(50)	YES		NULL	
Country	varchar(50)	YES		NULL	
IGN	varchar(50)	YES		NULL	
Email	varchar(50)	YES		NULL	
Pasword	varchar(50)	YES		NULL	
Varified	tinyint(1)	YES		NULL	
Fb_Auth	tinyint(1)	YES		NULL	
Google_Auth	tinyint(1)	YES		NULL	

Table name: log_in

Field	Type	Null	Key	Default	Extra
Login_Id	int	NO	PRI	NULL	auto_increment
User_Id	decimal(10,0)	YES	MUL	NULL	
Login_Time	timestamp	YES		CURRENT_TIMESTAMP	DEFAULT_GENERATED
Logout_Time	timestamp	YES		NULL	

Table name: character_stat

Field	Type	Null	Key	Default	Extra
Character_Id	decimal(10,0)	NO	PRI	NULL	
Character_Name	varchar(50)	YES		NULL	
User_Id	decimal(10,0)	YES	MUL	NULL	
Kills	decimal(10,0)	YES		NULL	
Headshot	decimal(10,0)	YES		NULL	
AR_Kill	decimal(10,0)	YES		NULL	
SMG_Kill	decimal(10,0)	YES		NULL	
LMG_Kill	decimal(10,0)	YES		NULL	
Pistol_Kill	decimal(10,0)	YES		NULL	
Win	decimal(10,0)	YES		NULL	
Damage	decimal(10,0)	YES		NULL	
Revive	decimal(10,0)	YES		NULL	
Game_played	decimal(10,0)	YES		NULL	
Shot_Gun	decimal(10,0)	YES		NULL	
Sniper_Kill	decimal(10,0)	YES		NULL	

Table name: multiplayer

Field	Type	Null	Key	Default	Extra
Match_Id	decimal(10,0)	NO	PRI	NULL	
User_Id	decimal(10,0)	NO	PRI	NULL	
Result_Id	decimal(10,0)	YES		NULL	
Team_Id	decimal(10,0)	YES		NULL	

Table name: summary

Field	Type	Null	Key	Default	Extra
Match_Id	decimal(10,0)	NO	PRI	NULL	
User_Id	decimal(10,0)	NO	PRI	NULL	
Kills	decimal(10,0)	YES		NULL	
Damage	decimal(10,0)	YES		NULL	
result_position	varchar(20)	YES		NULL	

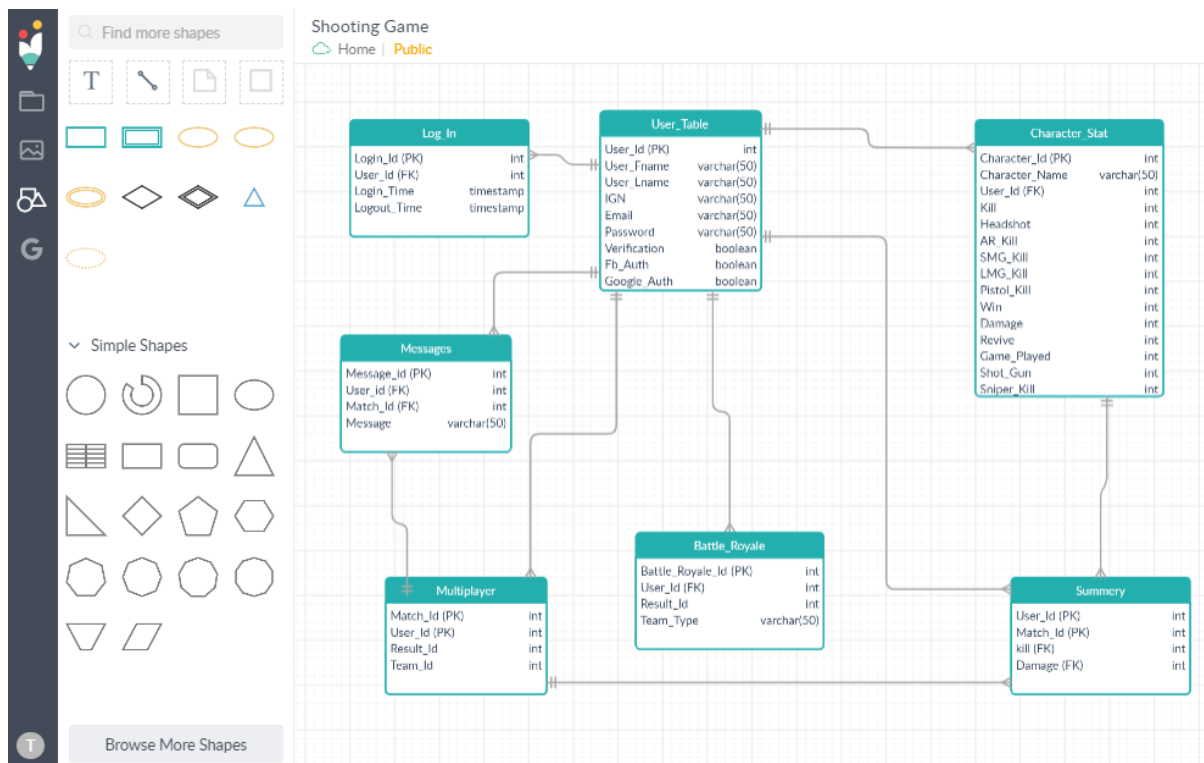
Table name: battle_royale

Field	Type	Null	Key	Default	Extra
Battle_Royale_Id	decimal(10,0)	NO	PRI	NULL	
User_Id	decimal(10,0)	NO	PRI	NULL	
Result_Id	decimal(10,0)	YES		NULL	
Team_Type	varchar(50)	YES		NULL	

Entity-Relationship Diagram:

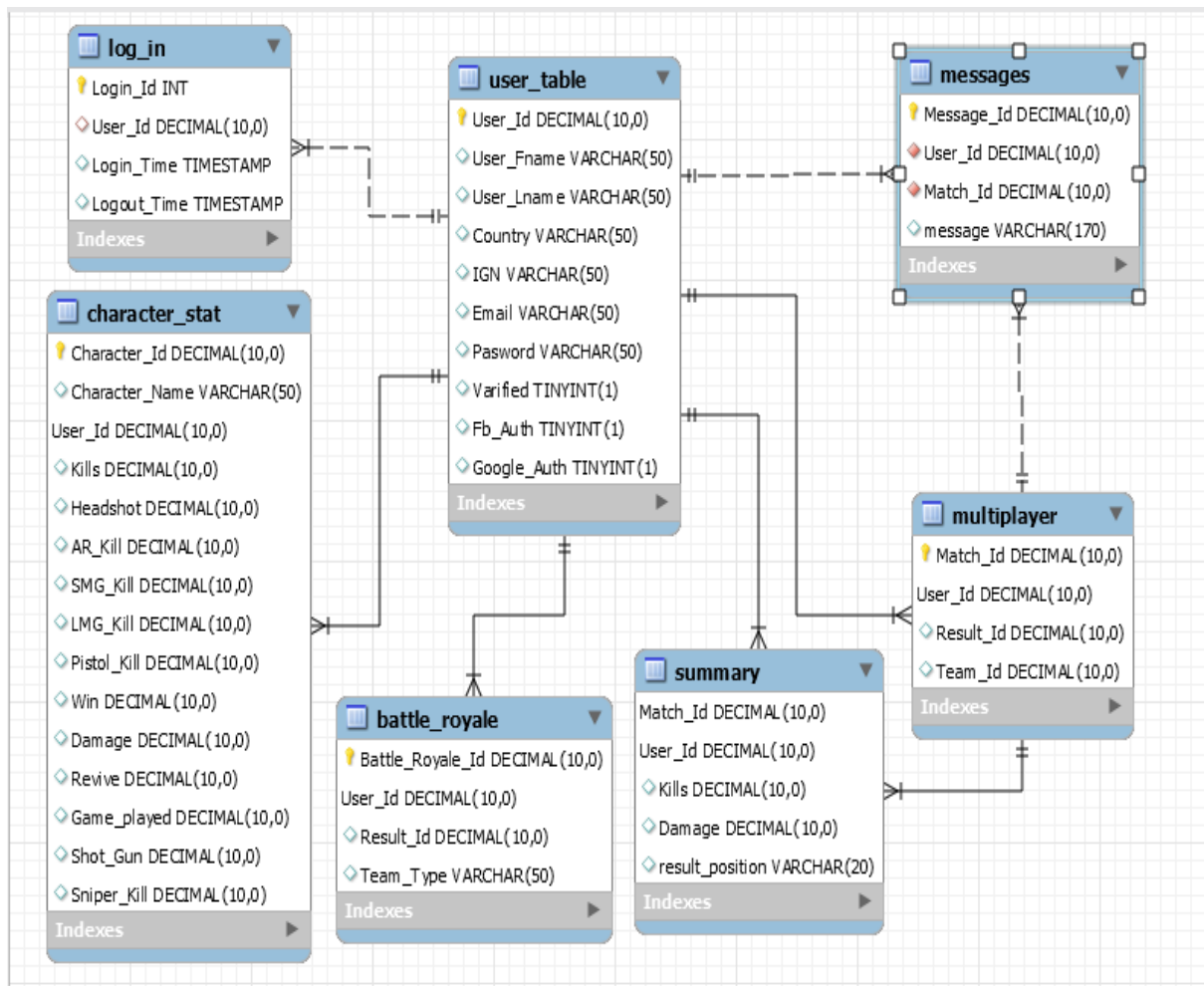
An entity-relationship (ER) diagram is a specialized graphic that illustrates the relationships between entities with all their attributes in a database. ER diagrams often use symbols to represent three different types of information. Boxes are commonly used to represent entities. Crow's foot model is used to represent the relationships between entities. it also shows the primary keys and foreign keys of the entities.

here is the ER diagram of our database we did with the help of this website - <https://app.creately.com/diagram/fk9TaGFL9WI/edit> :



Database Schema design:

MySQL Workbench simplifies database design and maintenance, automates time-consuming and error-prone tasks, and improves communication among DBA and developer teams. It enables data architects to visualize requirements, communicate with stakeholders, and resolve design issues before a major investment of time and resources is made. It enables model-driven database design, which is the most efficient methodology for creating valid and well-performing databases while providing the flexibility to respond to evolving business requirements. Model and Schema Validation utilities enforce best practice standards for data modeling, and also enforce MySQL-specific physical design standards so no mistakes are made when building new ER diagrams or generating physical MySQL databases.



Testing:

Database testing includes performing data validity, data integrity testing, performance check related to database and testing of procedures, triggers and functions in the database. It is very important to learn about DB testing and be able to validate Databases effectively to ensure security and quality databases.

suppose, an application that captures the day-to-day transaction details for users and stores the details in the database. From database testing point of view, the following checks should be performed –

- The transactional information from the application should be stored in the database and it should provide correct information to the user.
- Information should not be lost when it is loaded to the database.
- Only completed transactions should be stored and all incomplete operations should be aborted by the application.
- Access authorization to database should be maintained. No unapproved or unauthorized access to user information should be provided.

reasons for Database testing are–

- To ease the complexity of calls to the database backend, developers increase the use of View and Stored Procedures.
- These Stored procedures and Views contain critical tasks such as inserting customer details (name, contact information, etc.) and sales data. These tasks need to be tested at several levels.
- Black-box testing performed on front-end is important, but makes it difficult to isolate the problem. Testing at the backend system increases the robustness of the data. That is why database testing is performed on the back end system.
- In a database, data comes from multiple applications and there is a possibility that harmful or incorrect data is stored in the database. Therefore, there is a need to check database components regularly. In addition, data integrity and consistency should be checked regularly.

To test this database, we entered some data into the tables which we collected from some of our friends who regularly play this shooting game. These all data are given below:

user_table:

User_Id	User_Fname	User_Lname	Country	IGN	Email	Pasword	Varified	Fb_Auth	Google_Auth
111001	Salman	Alif	Bangladesh	xyzpn	SAlif@gmail	salif009	1	0	1
111002	Kayes	Fahim	Bangladesh	xyzsm	KFahim@gmail	KFahim501	1	1	1
111003	Tanzina	Meem	Bangladesh	xyzbb	Tanzina@gmail	Tanmeem777	1	1	1
111004	Anil	Hasan	India	xyzsm	AnilH@gmail	anil838	0	0	1
111005	Alex	Cooper	USA	abcmny	AlexK@gmail	alex007	1	0	1
111007	Sheldon	jackson	UK	abcbg	Shelly@gmail	sheldonJ0007	0	0	0
111009	James	Fraser	Scotland	abcr	Jamie@gmail	Fraser20	0	1	0
111011	Nick	Jason	USA	abcvv	Nicky@gmail	NickJ695	1	0	0
111013	Shabbar	Islam	Pakistan	xyzff	Shaon@gmail	ShaonI111	0	1	1
111015	Bella	Swan	USA	abcnm	SBella@gmail	Bellan088	1	1	0
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

log_in:

Login_Id	User_Id	Login_Time	Logout_Time
11	111013	2020-09-21 00:00:01	2020-09-21 04:50:33
91	111003	2020-01-11 12:55:27	2020-01-12 09:30:11
101	111001	2020-09-23 11:45:01	2020-09-24 13:20:57
111	111002	2020-09-25 14:52:49	2020-09-25 22:35:38
451	111007	2020-05-21 04:45:46	2020-05-21 09:20:13
452	111015	2020-09-26 16:27:22	NULL

character_stat:

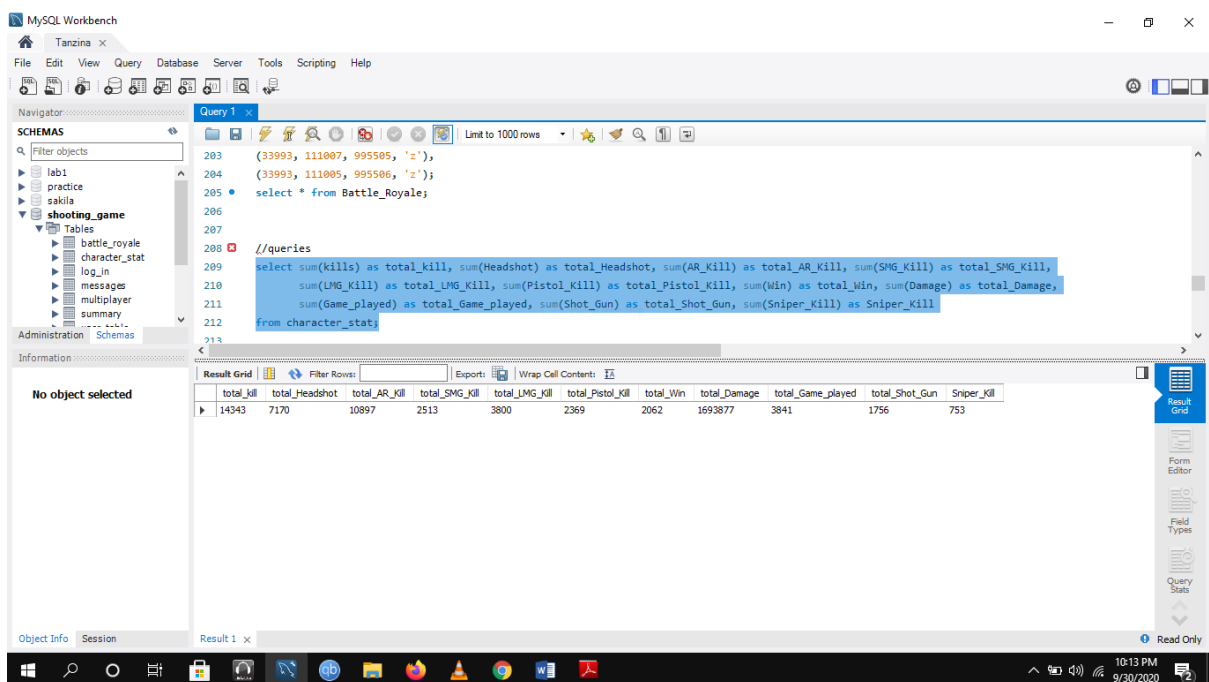
Character_Id	Character_Name	User_Id	Kills	Headshot	AR_Kill	SMG_Kill	LMG_Kill	Pistol_Kill	Win	Damage	Revive	Game_played	Shot_Gun	Sniper_Kill
991	Crypto	111001	14	16	23	44	33	46	48	5862	1	80	25	65
991	Crypto	111002	111	15	33	4	133	20	18	3672	12	43	35	34
991	Crypto	111003	13	19	9	12	7	21	6	5477	4	11	7	13
991	Crypto	111005	24	54	29	53	24	53	23	1678	13	32	43	23
991	Crypto	111007	34	15	39	32	37	11	36	4122	21	41	17	12
992	Wraith	111001	107	126	52	75	16	65	21	36311	2	57	11	22
992	Wraith	111002	76	156	33	15	36	43	24	45431	4	43	3	12
992	Wraith	111003	11	17	14	6	13	3	19	5434	5	21	17	6
992	Wraith	111005	321	42	653	536	33	23	43	3224	65	34	65	12
993	Bangalore	111001	3193	3680	211	322	154	555	654	1051231	886	678	345	77
993	Bangalore	111002	3223	654	2345	64	134	143	654	265431	654	642	652	27
993	Bangalore	111003	5422	564	123	422	1214	15	32	45632	211	42	22	12
993	Bangalore	111005	432	220	431	32	432	11	24	52322	36	55	76	111
993	Bangalore	111007	542	34	5433	35	77	543	33	4212	46	567	54	72
994	Pathfinder	111001	18	17	42	15	35	5	13	4737	9	19	25	35
994	Pathfinder	111002	32	432	32	12	52	11	21	5431	5	123	52	15
994	Pathfinder	111003	54	21	532	542	31	15	42	2343	6	532	66	33
994	Pathfinder	111005	33	432	22	42	42	22	43	5431	4	44	12	21
994	Pathfinder	111007	32	44	22	45	345	45	7	98764	23	49	8	9
995	Bloodhound	111001	90	118	73	13	41	53	72	29280	3	22	65	33
995	Bloodhound	111002	43	74	76	12	351	563	12	4565	2	43	55	22
995	Bloodhound	111003	8	10	19	2	5	7	16	3416	4	21	17	11
995	Bloodhound	111005	21	31	42	46	432	22	32	1234	1	521	44	22
995	Bloodhound	111007	442	22	42	22	15	4	22	422	42	1	3	5
997	Mirage	111001	3	1	8	6	14	7	6	1921	1	8	4	9
997	Mirage	111002	1	2	3	5	4	1	8	951	1	4	5	7
997	Mirage	111003	11	322	113	35	14	31	58	4311	11	43	25	24
997	Mirage	111007	32	32	443	64	76	31	75	1032	12	65	3	9

multiplayer:

Match_Id	User_Id	Result_Id	Team_Id
101	111001	997701	551
101	111002	997702	551
101	111003	997703	551
103	111005	997704	553
103	111007	997705	553
103	111013	997707	553

Then we created some query questions and solved them using sql to test the database. Here are some query questions and the answers that showed in mysql.

1. find out the total count of every kind of kill in the Character_stat table.



The screenshot shows the MySQL Workbench interface. The 'Query Editor' window contains the following SQL code:

```

(33993, 111007, 995505, 'z'),
(33993, 111005, 995506, 'z');
select * from Battle_Royale;

//queries
select sum(kills) as total_kill, sum(Headshot) as total_Headshot, sum(AR_Kill) as total_AR_Kill, sum(SMG_Kill) as total_SMG_Kill,
sum(LMG_Kill) as total_LMG_Kill, sum(Pistol_Kill) as total_Pistol_Kill, sum(Win) as total_Win, sum(Damage) as total_Damage,
sum(Game_played) as total_Game_played, sum(Shot_Gun) as total_Shot_Gun, sum(Sniper_Kill) as Sniper_Kill
from character_stat;

```

The 'Result Grid' window displays the following data:

total_kill	total_Headshot	total_AR_Kill	total_SMG_Kill	total_LMG_Kill	total_Pistol_Kill	total_Win	total_Damage	total_Game_played	total_Shot_Gun	Sniper_Kill
14343	7170	10897	2513	3800	2369	2062	1693877	3841	1756	753

2. find out total kill and total win of every player.

The screenshot shows MySQL Workbench with a query editor and a result grid. The query is as follows:

```

212 from character_stat;
213
214 • select u.user_id, u.User_Fname, u.User_Lname, sum(c.kills) as total_kill, sum(c.win) as total_win
215 from user_table u
216 left outer join character_stat c
217 on u.user_id = c.user_id
218 group by User_Id;
219
220 • select Character_Name, Character_Id, sum(kills) as total_kill, sum(win) as total_win
221 from character_stat
222 group by Character_Name;

```

The result grid displays the following data:

user_id	User_Fname	User_Lname	total_kill	total_win
111001	Salman	Alif	3425	814
111002	Kayes	Fahim	3486	737
111003	Tanzina	Meem	5519	173
111004	Anil	Hasan	NULL	NULL
111005	Alex	Cooper	831	165
111007	Sheldon	Jackson	1082	173
111009	James	Fraser	NULL	NULL
111011	Nick	Jason	NULL	NULL
111013	Shabbir	Islam	NULL	NULL
111015	Bela	Swan	NULL	NULL

3. find out who has the most kills.

The screenshot shows MySQL Workbench with a query editor and a result grid. The query is as follows:

```

227 on u.user_id = c.user_id
228 group by User_Id;
229
230 • select user_id, Max_kill
231 from (select user_id, sum(kills) as Max_kill
232 from character_stat
233 group by User_Id) derived_table_name
234 group by User_Id
235 order by Max_kill desc
236 limit 1;

```

The result grid displays the following data:

user_id	Max_kill
111003	5519

4. find out the activities of every player in "Multiplayer" game mode - like their messages, results etc.

MySQL Workbench

Tanzina x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

lab1

practice

sakila

shooting_game

Tables

battle_royale

character_stat

log_in

messages

multiplayer

summary

Administration Schemas

Information

No object selected

Query 1

```

253
254 • select u.User_Fname, u.User_Lname, u.User_Id, mp.Match_Id, mp.Team_Id, ms.message, s.Kills, s.Damage, s.result_position
255 from user_table u
256 left outer join multiplayer mp
257 on (u.User_Id = mp.User_Id)
258 left outer join messages ms
259 on (mp.Match_Id = ms.Match_Id and ms.User_Id = u.User_Id)
260 left outer join summary s
261 on (s.Match_Id = mp.Match_Id and s.User_Id = u.User_Id);
262
263 • select u.User_Fname, u.User_Lname, u.User_Id, mp.Match_Id, mp.Team_Id, ms.message, s.Kills, s.Damage, s.result_position

```

Result Grid

Filter Rows: Exports Wrap Cell Contents

User_Fname	User_Lname	User_Id	Match_Id	Team_Id	message	Kills	Damage	result_position
Salman	Alif	111001	101	551	Hello	12	7	1st
Kayes	Fahim	111002	101	551	whats up?	23	18	1st
Tanzina	Meem	111003	101	551	we guys r playing really well today.	12	7	1st
Anil	Hasan	111004	NAKE	NAKE	NAKE	NAKE	NAKE	NAKE
Alex	Cooper	111005	103	553	there is an enemy.	9	17	4th
Sheldon	Jackson	111007	103	553	yeah, i got it. dont worry	5	11	4th
James	Fraser	111009	NAKE	NAKE	NAKE	NAKE	NAKE	NAKE
Nick	Jason	111011	NAKE	NAKE	NAKE	NAKE	NAKE	NAKE
Shabbar	Islam	111013	103	553	overconfident....you missed. haha...	10	9	4th
Bella	Swan	111015	NAKE	NAKE	NAKE	NAKE	NAKE	NAKE

Object Info Session Result 5 x

Read Only

10:17 PM 9/30/2020

6. Discussion:

This is a database project without any front and back end. Although During making this project, we faced some problems. Once we build a table, we can not change it later at our own will. we had to first drop the already made table, then make the changes. making foreign keys and linking them to other tables took some getting used to and whether we should set the foreign keys on null or cascade - that created some confusion. During testing our database, we ran some queries. and when we wanted to find the player who had the highest total kill, we needed the nested aggregate function - `max(sum(column))` which did not work. otherwise we did not face any more problems.

7. Conclusion

So basically, we built a Database that stores player data, player game stats and game score. For supposedly using this database shooting game can be managed successfully .

Unfortunately, we couldn't fulfill our claim that we had made in the beginning of the semester, we had a lot of plans. We will implement more tables and datasets for more game mood, playing with friends and character cosmetics . We firmly believe that this database will help game developer companies to manage their players. It will be far more beneficial in the future when we really need some information.

8. Acknowledgement

First of all, we are grateful to the Almighty that we've been able to complete this project on time.

We would like to take this opportunity to express our sincere gratitude to our honorable course instructor, **Kamruddin Md. Nur**, for his intellectual guidance throughout the course. The knowledge that has been imparted will go a long way in solving future critical conditions and also it will help us all throughout our professional career. Without his immense help we wouldn't have been able to complete this report.

This report cannot be solely attributed to only one's effort, but it is indeed the joint effort of all the fellow members of the group. There were hard times during the preparation of this report but it could not have been possible to write it without the solely hard work help of all the group members.

9. References

1. Dino Anastasia, Marla Gómez. (2004). Final Project Report - SI654: Database Application Design.
2. <https://www.studentprojectguide.com/project-report/database-design/electricity-billing-system-database-design/>
3. https://www.tutorialspoint.com/database_testing/database_testing_overview.htm
- 4.