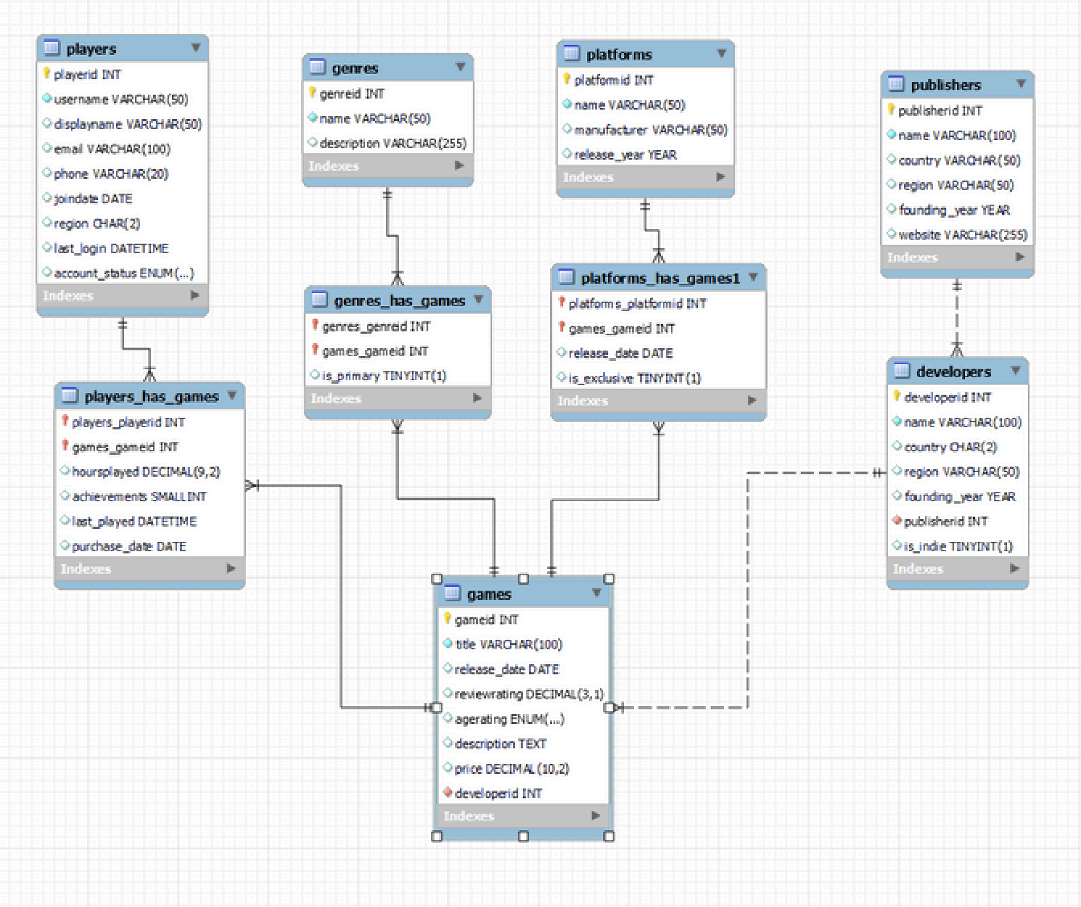
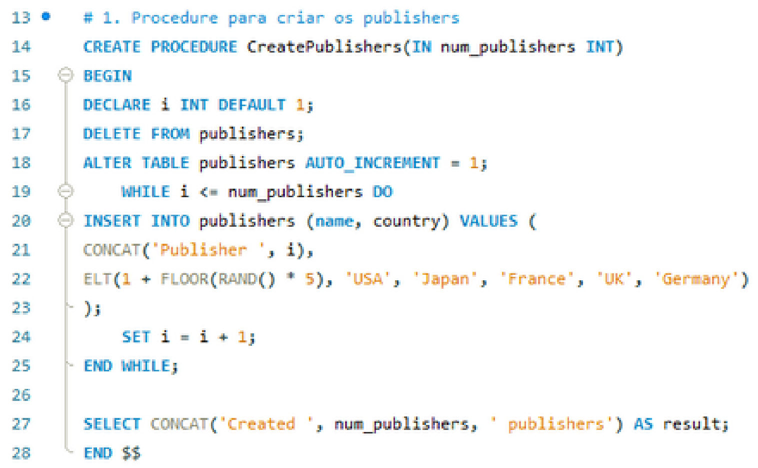
# Matheus Braga Cetrangolo - CC3Mb

https://github.com/Kvyotho/BD2-CC3MB-Projeto02

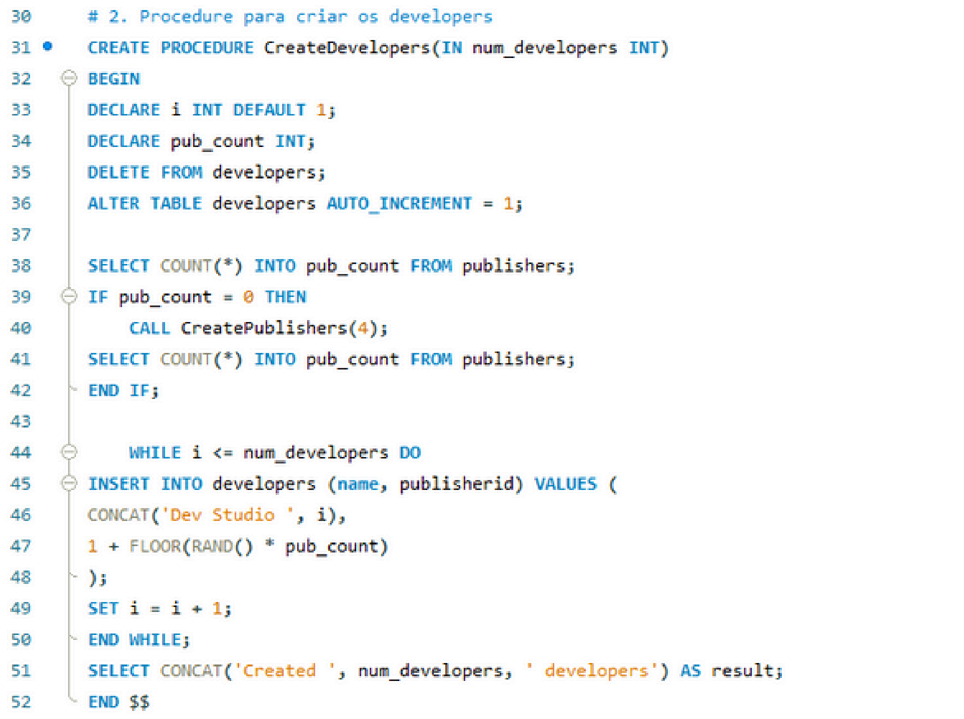
## 1. Arquitetura Banco de Dados



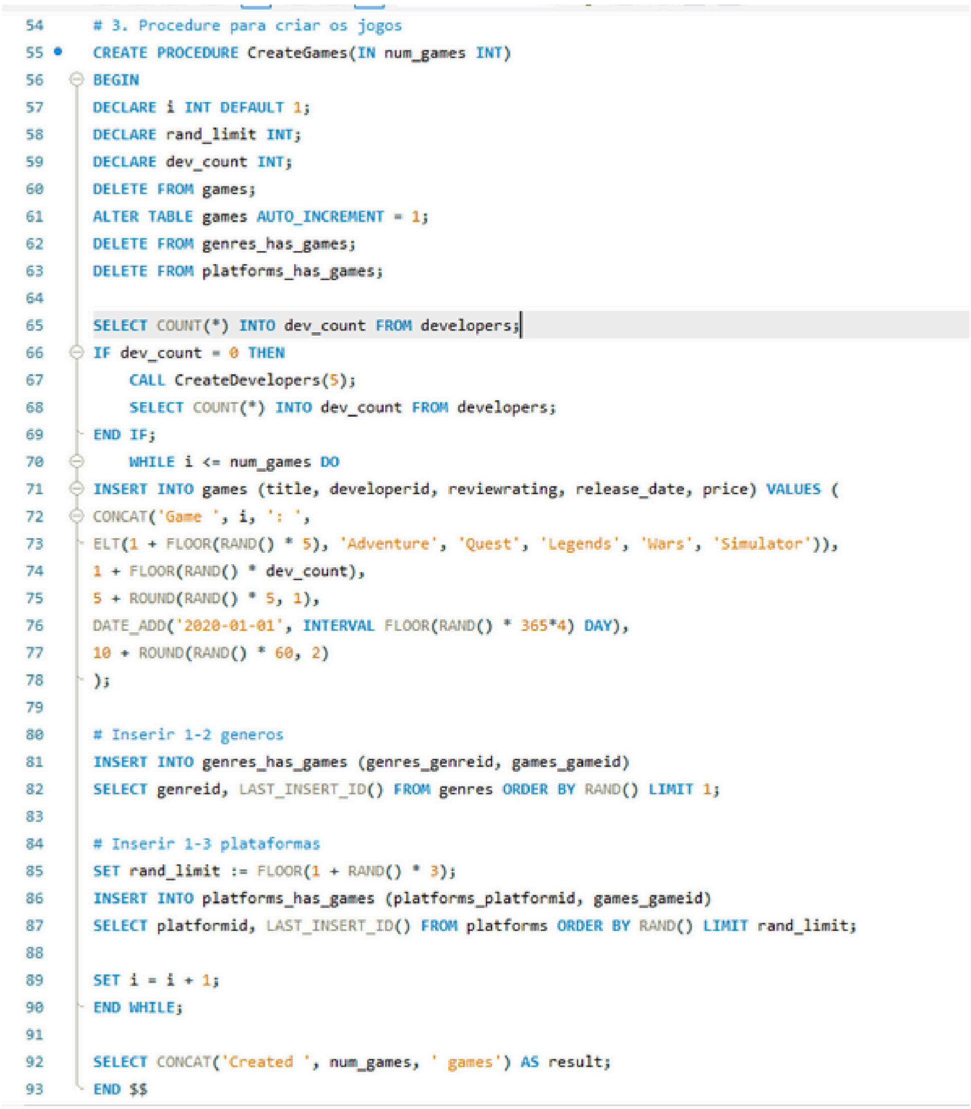
## 2. Popular banco de dados



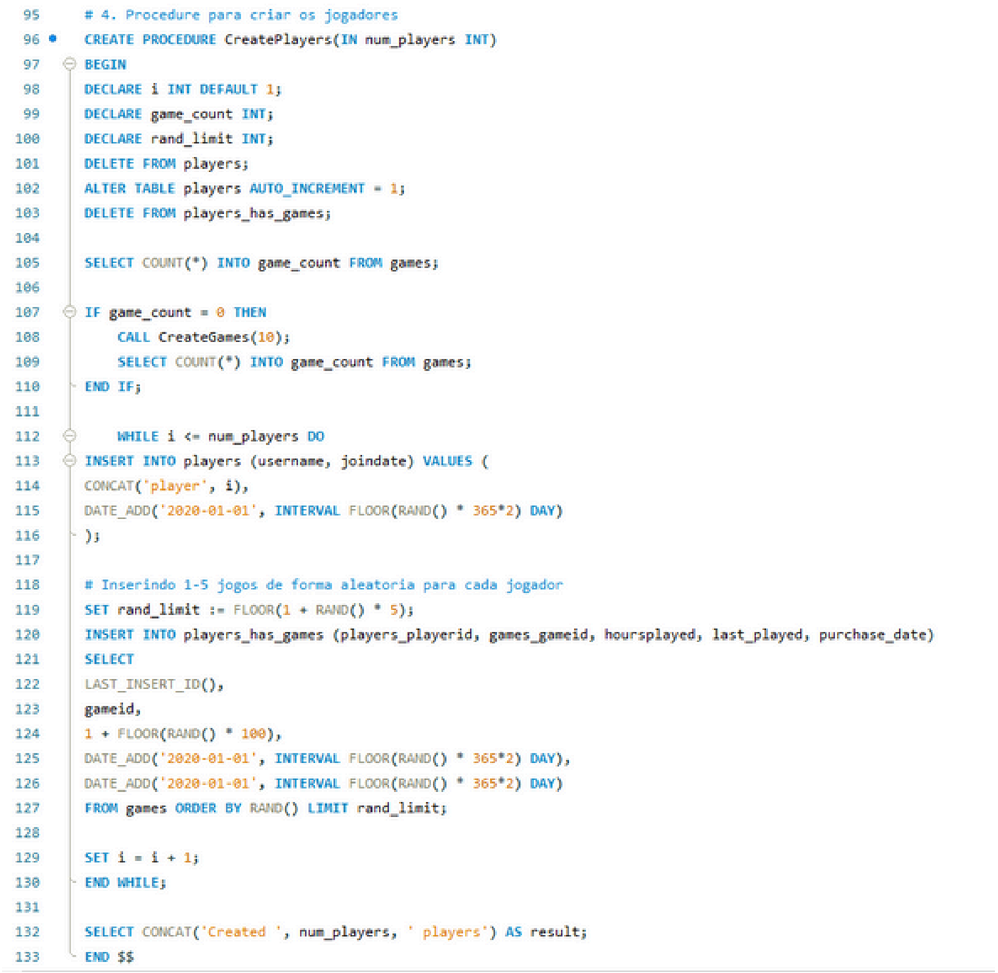
```sql
# 1. Procedure para criar os publishers
CREATE PROCEDURE CreatePublishers(IN num_publishers INT)
BEGIN
DECLARE i INT DEFAULT 1;
DELETE FROM publishers;
ALTER TABLE publishers AUTO_INCREMENT = 1;
    WHILE i <= num_publishers DO
INSERT INTO publishers (name, country) VALUES (
CONCAT('Publisher ', i),
ELT(1 + FLOOR(RAND() * 5), 'USA', 'Japan', 'France', 'UK', 'Germany')
);
    SET i = i + 1;
END WHILE;

SELECT CONCAT('Created ', num_publishers, ' publishers') AS result;
END $$
```



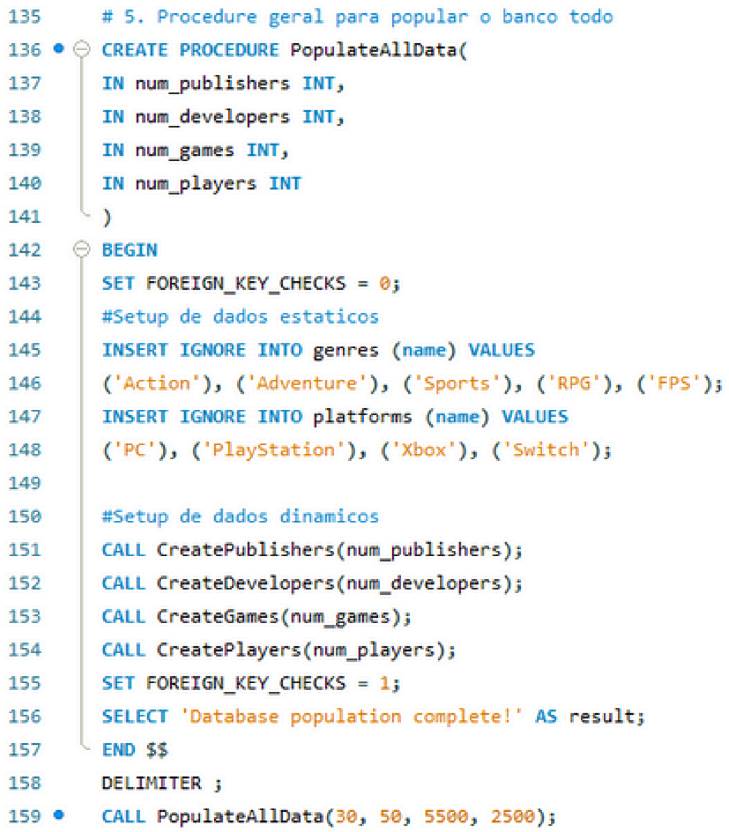
```sql
# 2. Procedure para criar os developers
CREATE PROCEDURE CreateDevelopers(IN num_developers INT)
BEGIN
DECLARE i INT DEFAULT 1;
DECLARE pub_count INT;
DELETE FROM developers;
ALTER TABLE developers AUTO_INCREMENT = 1;

SELECT COUNT(*) INTO pub_count FROM publishers;
IF pub_count = 0 THEN
    CALL CreatePublishers(4);
SELECT COUNT(*) INTO pub_count FROM publishers;
END IF;

    WHILE i <= num_developers DO
INSERT INTO developers (name, publisherid) VALUES (
CONCAT('Dev Studio ', i),
1 + FLOOR(RAND() * pub_count)
);
SET i = i + 1;
END WHILE;
SELECT CONCAT('Created ', num_developers, ' developers') AS result;
END $$
```



```sql
# 3. Procedure para criar os jogos
CREATE PROCEDURE CreateGames(IN num_games INT)
BEGIN
DECLARE i INT DEFAULT 1;
DECLARE rand_limit INT;
DECLARE dev_count INT;
DELETE FROM games;
ALTER TABLE games AUTO_INCREMENT = 1;
DELETE FROM genres_has_games;
DELETE FROM platforms_has_games;

SELECT COUNT(*) INTO dev_count FROM developers;
IF dev_count = 0 THEN
    CALL CreateDevelopers(5);
    SELECT COUNT(*) INTO dev_count FROM developers;
END IF;
    WHILE i <= num_games DO
INSERT INTO games (title, developerid, reviewrating, release_date, price) VALUES (
CONCAT('Game ', i, ': ',
ELT(1 + FLOOR(RAND() * 5), 'Adventure', 'Quest', 'Legends', 'Wars', 'Simulator')),
1 + FLOOR(RAND() * dev_count),
5 + ROUND(RAND() * 5, 1),
DATE_ADD('2020-01-01', INTERVAL FLOOR(RAND() * 365*4) DAY),
10 + ROUND(RAND() * 60, 2)
);

# Inserir 1-2 generos
INSERT INTO genres_has_games (genres_genreid, games_gameid)
SELECT genreid, LAST_INSERT_ID() FROM genres ORDER BY RAND() LIMIT 1;

# Inserir 1-3 plataformas
SET rand_limit := FLOOR(1 + RAND() * 3);
INSERT INTO platforms_has_games (platforms_platformid, games_gameid)
SELECT platformid, LAST_INSERT_ID() FROM platforms ORDER BY RAND() LIMIT rand_limit;

SET i = i + 1;
END WHILE;

SELECT CONCAT('Created ', num_games, ' games') AS result;
END $$
```

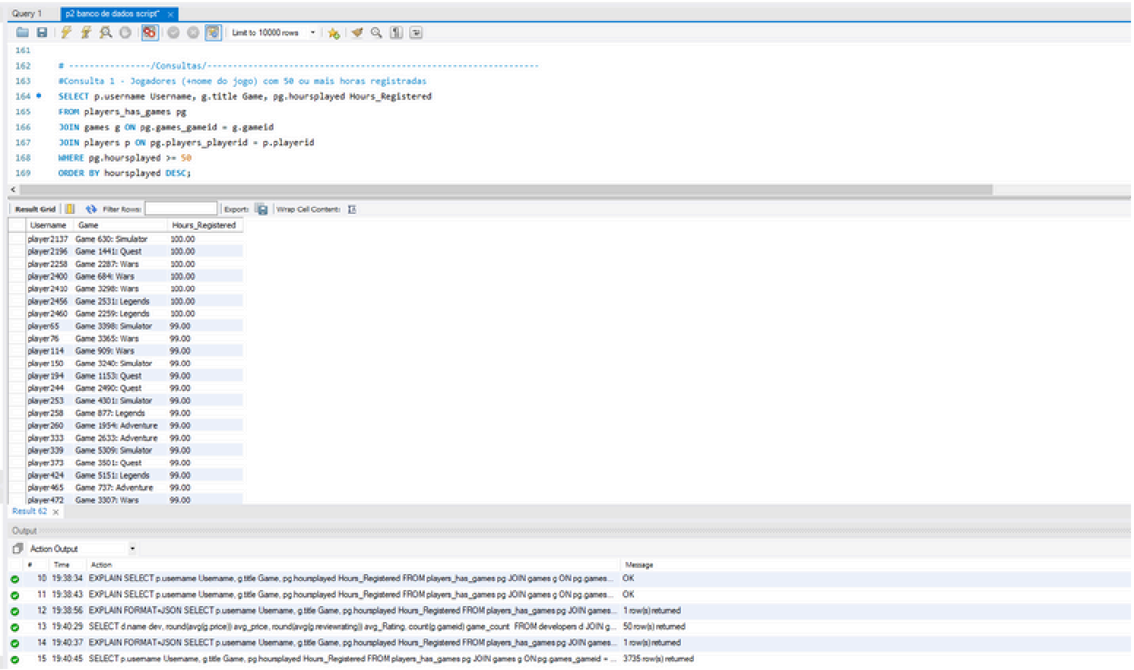


```sql
# 4. Procedure para criar os jogadores
CREATE PROCEDURE CreatePlayers(IN num_players INT)
BEGIN
DECLARE i INT DEFAULT 1;
DECLARE game_count INT;
DECLARE rand_limit INT;
DELETE FROM players;
ALTER TABLE players AUTO_INCREMENT = 1;
DELETE FROM players_has_games;

SELECT COUNT(*) INTO game_count FROM games;

IF game_count = 0 THEN
    CALL CreateGames(10);
    SELECT COUNT(*) INTO game_count FROM games;
END IF;

    WHILE i <= num_players DO
INSERT INTO players (username, joindate) VALUES (
CONCAT('player', i),
DATE_ADD('2020-01-01', INTERVAL FLOOR(RAND() * 365*2) DAY)
);

# Inserindo 1-5 jogos de forma aleatoria para cada jogador
SET rand_limit := FLOOR(1 + RAND() * 5);
INSERT INTO players_has_games (players_playerid, games_gameid, hoursplayed, last_played, purchase_date)
SELECT
LAST_INSERT_ID(),
gameid,
1 + FLOOR(RAND() * 100),
DATE_ADD('2020-01-01', INTERVAL FLOOR(RAND() * 365*2) DAY),
DATE_ADD('2020-01-01', INTERVAL FLOOR(RAND() * 365*2) DAY)
FROM games ORDER BY RAND() LIMIT rand_limit;

SET i = i + 1;
END WHILE;

SELECT CONCAT('Created ', num_players, ' players') AS result;
END $$
```



```sql
# 5. Procedure geral para popular o banco todo
CREATE PROCEDURE PopulateAllData(
IN num_publishers INT,
IN num_developers INT,
IN num_games INT,
IN num_players INT
)
BEGIN
SET FOREIGN_KEY_CHECKS = 0;
#Setup de dados estaticos
INSERT IGNORE INTO genres (name) VALUES
('Action'), ('Adventure'), ('Sports'), ('RPG'), ('FPS');
INSERT IGNORE INTO platforms (name) VALUES
('PC'), ('PlayStation'), ('Xbox'), ('Switch');

#Setup de dados dinamicos
CALL CreatePublishers(num_publishers);
CALL CreateDevelopers(num_developers);
CALL CreateGames(num_games);
CALL CreatePlayers(num_players);
SET FOREIGN_KEY_CHECKS = 1;
SELECT 'Database population complete!' AS result;
END $$
DELIMITER ;
CALL PopulateAllData(30, 50, 5500, 2500);
```

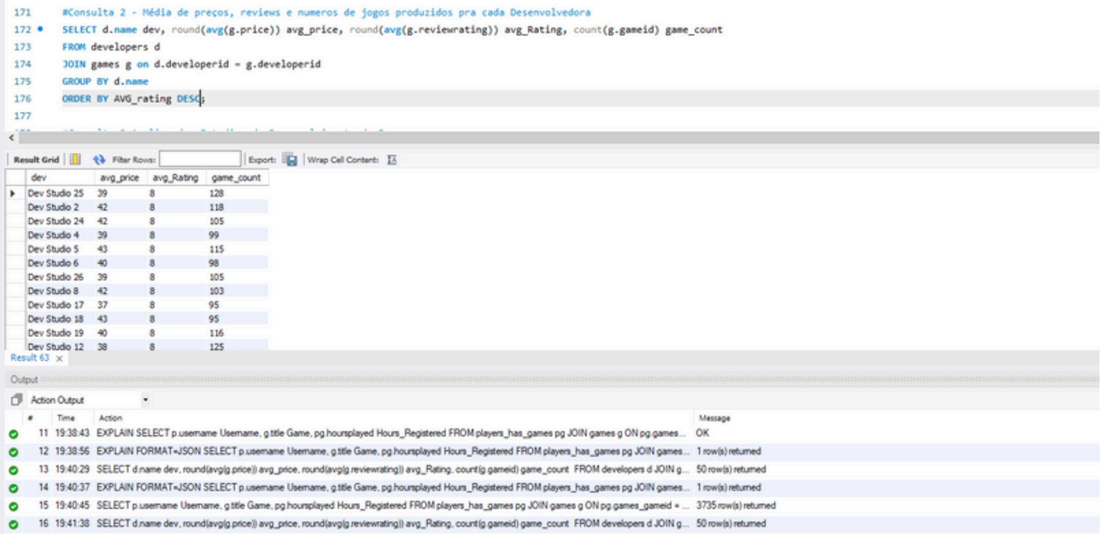
# 3. Consultas



```sql
# ----------------/Consultas/--------------------------------------------------------------
#Consulta 1 - Jogadores (+nome do jogo) com 50 ou mais horas registradas
SELECT p.username Username, g.title Game, pg.hoursplayed Hours_Registered
FROM players_has_games pg
JOIN games g ON pg.games_gameid = g.gameid
JOIN players p ON pg.players_playerid = p.playerid
WHERE pg.hoursplayed >= 50
ORDER BY hoursplayed DESC;
```

| Username | Game | Hours_Registered |
|---|---|---|
| player2137 | Game 630: Simulator | 100.00 |
| player2196 | Game 1441: Quest | 100.00 |
| player2258 | Game 2287: Wars | 100.00 |
| player2400 | Game 684: Wars | 100.00 |
| player2410 | Game 3298: Wars | 100.00 |
| player2456 | Game 2531: Legends | 100.00 |
| player2460 | Game 2259: Legends | 100.00 |
| player65 | Game 3398: Simulator | 99.00 |
| player76 | Game 3365: Wars | 99.00 |
| player114 | Game 909: Wars | 99.00 |
| player150 | Game 3240: Simulator | 99.00 |
| player194 | Game 1153: Quest | 99.00 |
| player244 | Game 2490: Quest | 99.00 |
| player253 | Game 4301: Simulator | 99.00 |
| player258 | Game 877: Legends | 99.00 |
| player260 | Game 1954: Adventure | 99.00 |
| player333 | Game 2633: Adventure | 99.00 |
| player339 | Game 5309: Simulator | 99.00 |
| player373 | Game 3501: Quest | 99.00 |
| player424 | Game 5151: Legends | 99.00 |
| player465 | Game 737: Adventure | 99.00 |
| player472 | Game 3307: Wars | 99.00 |

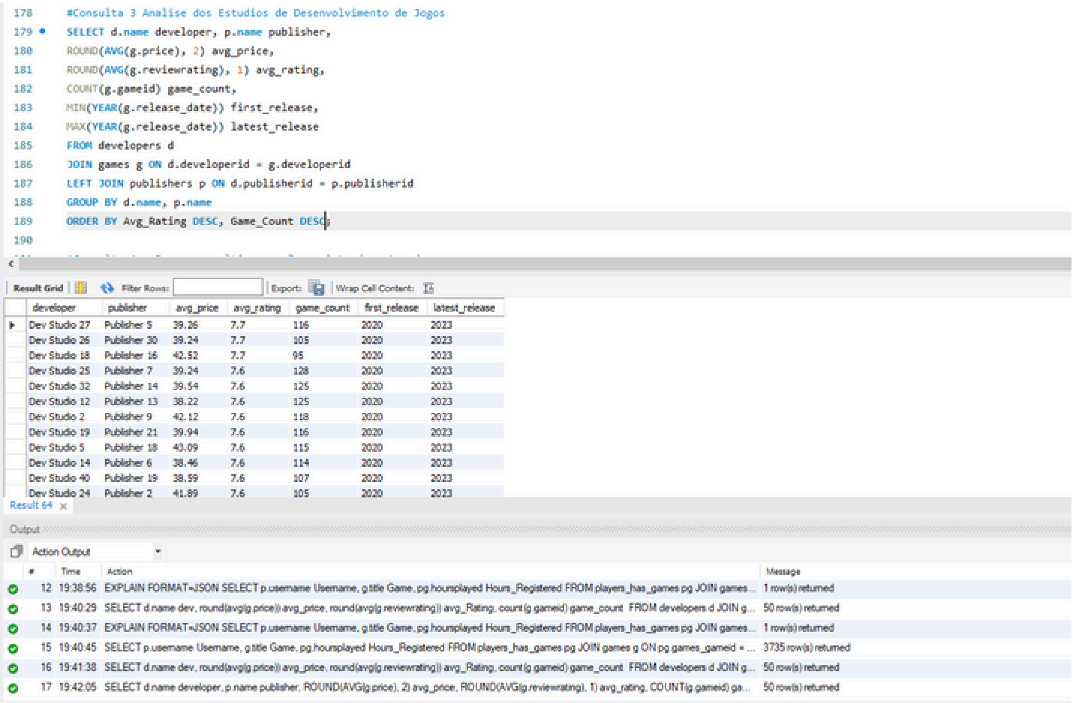
```
#   Time      Action                                                                                                                          Message
10  19:38:34  EXPLAIN SELECT p.username Username, g.title Game, pg.hoursplayed Hours_Registered FROM players_has_games pg JOIN games g ON pg.games...   OK
11  19:38:43  EXPLAIN SELECT p.username Username, g.title Game, pg.hoursplayed Hours_Registered FROM players_has_games pg JOIN games g ON pg.games...   OK
12  19:38:56  EXPLAIN FORMAT=JSON SELECT p.username Username, g.title Game, pg.hoursplayed Hours_Registered FROM players_has_games pg JOIN games...   1 row(s) returned
13  19:40:29  SELECT d.name dev, round(avg(g.price)) avg_price, round(avg(g.reviewrating)) avg_Rating, count(g.gameid) game_count  FROM developers d JOIN g...   50 row(s) returned
14  19:40:37  EXPLAIN FORMAT=JSON SELECT p.username Username, g.title Game, pg.hoursplayed Hours_Registered FROM players_has_games pg JOIN games...   1 row(s) returned
15  19:40:45  SELECT p.username Username, g.title Game, pg.hoursplayed Hours_Registered FROM players_has_games pg JOIN games g ON pg.games_gameid = ...   3735 row(s) returned
```



```sql
#Consulta 2 - Média de preços, reviews e numeros de jogos produzidos pra cada Desenvolvedora
SELECT d.name dev, round(avg(g.price)) avg_price, round(avg(g.reviewrating)) avg_Rating, count(g.gameid) game_count
FROM developers d
JOIN games g on d.developerid = g.developerid
GROUP BY d.name
ORDER BY AVG_rating DESC;
```

| dev | avg_price | avg_Rating | game_count |
|---|---|---|---|
| Dev Studio 25 | 39 | 8 | 128 |
| Dev Studio 2 | 42 | 8 | 118 |
| Dev Studio 24 | 42 | 8 | 105 |
| Dev Studio 4 | 39 | 8 | 99 |
| Dev Studio 5 | 43 | 8 | 115 |
| Dev Studio 6 | 40 | 8 | 98 |
| Dev Studio 26 | 39 | 8 | 105 |
| Dev Studio 8 | 42 | 8 | 103 |
| Dev Studio 17 | 37 | 8 | 95 |
| Dev Studio 18 | 43 | 8 | 95 |
| Dev Studio 19 | 40 | 8 | 116 |
| Dev Studio 12 | 38 | 8 | 125 |

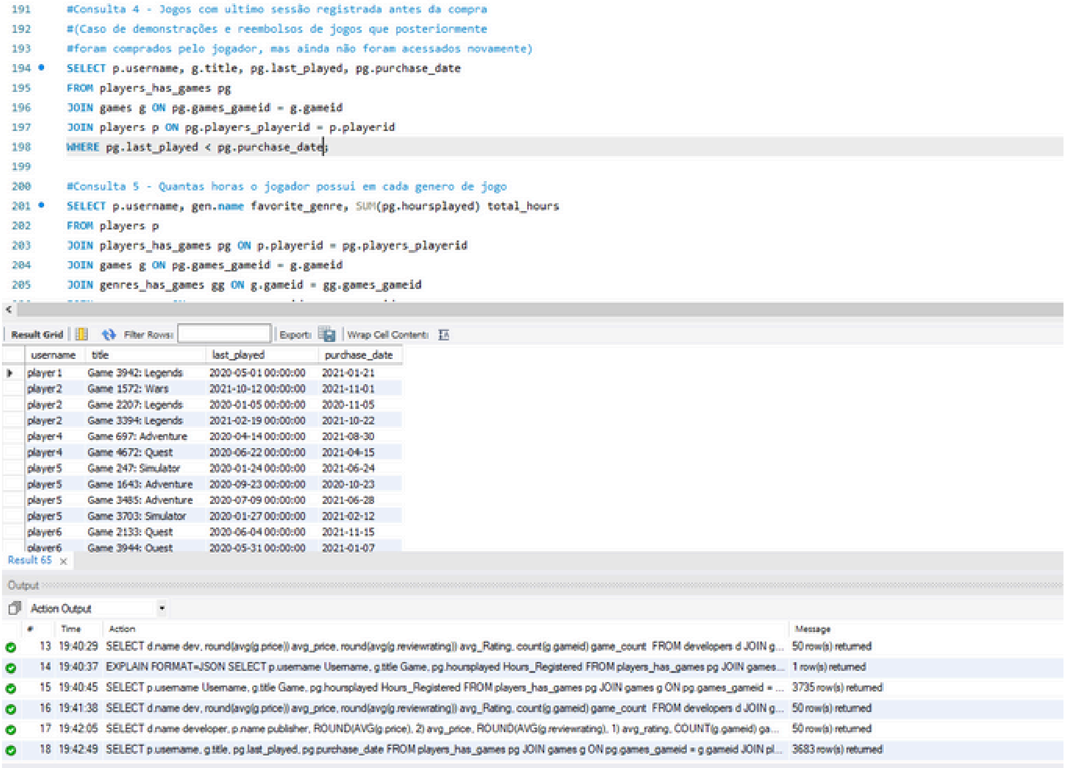
```
#   Time      Action                                                                                                                          Message
11  19:38:43  EXPLAIN SELECT p.username Username, g.title Game, pg.hoursplayed Hours_Registered FROM players_has_games pg JOIN games g ON pg.games...   OK
12  19:38:56  EXPLAIN FORMAT=JSON SELECT p.username Username, g.title Game, pg.hoursplayed Hours_Registered FROM players_has_games pg JOIN games...   1 row(s) returned
13  19:40:29  SELECT d.name dev, round(avg(g.price)) avg_price, round(avg(g.reviewrating)) avg_Rating, count(g.gameid) game_count  FROM developers d JOIN g...   50 row(s) returned
14  19:40:37  EXPLAIN FORMAT=JSON SELECT p.username Username, g.title Game, pg.hoursplayed Hours_Registered FROM players_has_games pg JOIN games...   1 row(s) returned
15  19:40:45  SELECT p.username Username, g.title Game, pg.hoursplayed Hours_Registered FROM players_has_games pg JOIN games g ON pg.games_gameid = ...   3735 row(s) returned
16  19:41:38  SELECT d.name dev, round(avg(g.price)) avg_price, round(avg(g.reviewrating)) avg_Rating, count(g.gameid) game_count  FROM developers d JOIN g...   50 row(s) returned
```



```sql
#Consulta 3 Analise dos Estudios de Desenvolvimento de Jogos
SELECT d.name developer, p.name publisher,
ROUND(AVG(g.price), 2) avg_price,
ROUND(AVG(g.reviewrating), 1) avg_rating,
COUNT(g.gameid) game_count,
MIN(YEAR(g.release_date)) first_release,
MAX(YEAR(g.release_date)) latest_release
FROM developers d
JOIN games g ON d.developerid = g.developerid
LEFT JOIN publishers p ON d.publisherid = p.publisherid
GROUP BY d.name, p.name
ORDER BY Avg_Rating DESC, Game_Count DESC;
```

| developer | publisher | avg_price | avg_rating | game_count | first_release | latest_release |
|---|---|---|---|---|---|---|
| Dev Studio 27 | Publisher 5 | 39.26 | 7.7 | 116 | 2020 | 2023 |
| Dev Studio 26 | Publisher 30 | 39.24 | 7.7 | 105 | 2020 | 2023 |
| Dev Studio 18 | Publisher 16 | 42.52 | 7.7 | 95 | 2020 | 2023 |
| Dev Studio 25 | Publisher 7 | 39.24 | 7.6 | 128 | 2020 | 2023 |
| Dev Studio 32 | Publisher 14 | 39.54 | 7.6 | 125 | 2020 | 2023 |
| Dev Studio 12 | Publisher 13 | 38.22 | 7.6 | 125 | 2020 | 2023 |
| Dev Studio 2 | Publisher 9 | 42.12 | 7.6 | 118 | 2020 | 2023 |
| Dev Studio 19 | Publisher 21 | 39.94 | 7.6 | 116 | 2020 | 2023 |
| Dev Studio 5 | Publisher 18 | 43.09 | 7.6 | 115 | 2020 | 2023 |
| Dev Studio 14 | Publisher 6 | 38.46 | 7.6 | 114 | 2020 | 2023 |
| Dev Studio 40 | Publisher 19 | 38.59 | 7.6 | 107 | 2020 | 2023 |
| Dev Studio 24 | Publisher 2 | 41.89 | 7.6 | 105 | 2020 | 2023 |

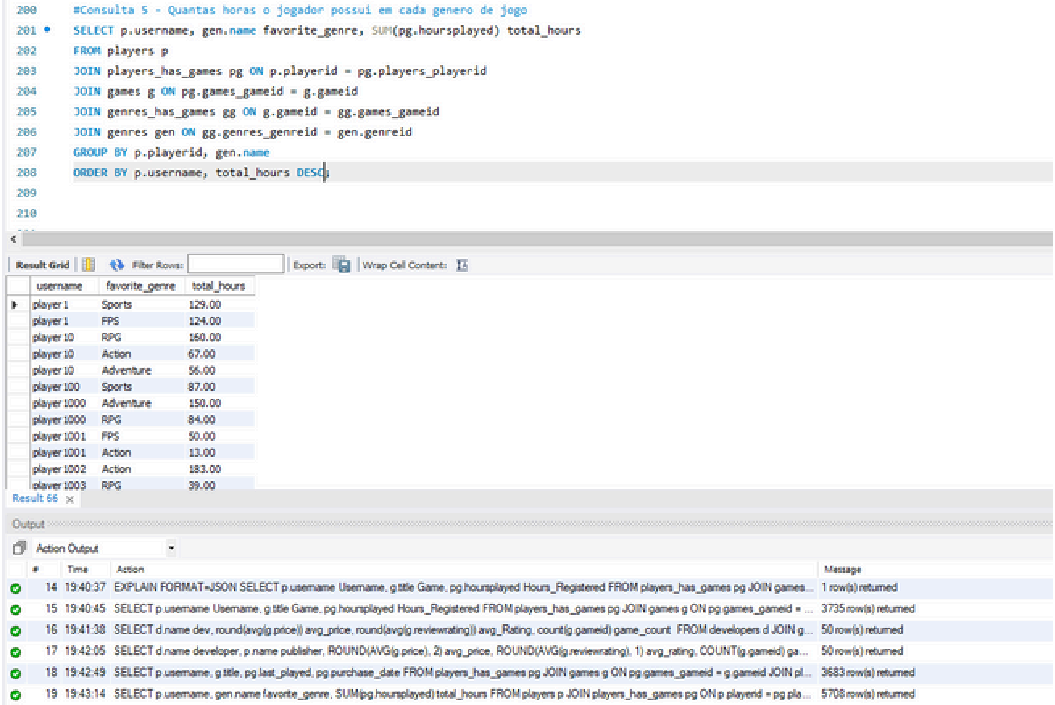
```
#   Time      Action                                                                                                                          Message
12  19:38:56  EXPLAIN FORMAT=JSON SELECT p.username Username, g.title Game, pg.hoursplayed Hours_Registered FROM players_has_games pg JOIN games...   1 row(s) returned
13  19:40:29  SELECT d.name dev, round(avg(g.price)) avg_price, round(avg(g.reviewrating)) avg_Rating, count(g.gameid) game_count  FROM developers d JOIN g...   50 row(s) returned
14  19:40:37  EXPLAIN FORMAT=JSON SELECT p.username Username, g.title Game, pg.hoursplayed Hours_Registered FROM players_has_games pg JOIN games...   1 row(s) returned
15  19:40:45  SELECT p.username Username, g.title Game, pg.hoursplayed Hours_Registered FROM players_has_games pg JOIN games g ON pg.games_gameid = ...   3735 row(s) returned
16  19:41:38  SELECT d.name dev, round(avg(g.price)) avg_price, round(avg(g.reviewrating)) avg_Rating, count(g.gameid) game_count  FROM developers d JOIN g...   50 row(s) returned
17  19:42:05  SELECT d.name developer, p.name publisher, ROUND(AVG(g.price), 2) avg_price, ROUND(AVG(g.reviewrating), 1) avg_rating, COUNT(g.gameid) ga...   50 row(s) returned
```



```sql
#Consulta 4 - Jogos com ultimo sessão registrada antes da compra
#(Caso de demonstrações e reembolsos de jogos que posteriormente
#foram comprados pelo jogador, mas ainda não foram acessados novamente)
SELECT p.username, g.title, pg.last_played, pg.purchase_date
FROM players_has_games pg
JOIN games g ON pg.games_gameid = g.gameid
JOIN players p ON pg.players_playerid = p.playerid
WHERE pg.last_played < pg.purchase_date;

#Consulta 5 - Quantas horas o jogador possui em cada genero de jogo
SELECT p.username, gen.name favorite_genre, SUM(pg.hoursplayed) total_hours
FROM players p
JOIN players_has_games pg ON p.playerid = pg.players_playerid
JOIN games g ON pg.games_gameid = g.gameid
JOIN genres_has_games gg ON g.gameid = gg.games_gameid
```

| username | title | last_played | purchase_date |
|---|---|---|---|
| player1 | Game 3942: Legends | 2020-05-01 00:00:00 | 2021-01-21 |
| player2 | Game 1572: Wars | 2021-10-12 00:00:00 | 2021-11-01 |
| player2 | Game 2207: Legends | 2020-01-05 00:00:00 | 2020-11-05 |
| player2 | Game 3394: Legends | 2021-02-19 00:00:00 | 2021-10-22 |
| player4 | Game 697: Adventure | 2020-04-14 00:00:00 | 2021-08-30 |
| player4 | Game 4672: Quest | 2020-06-22 00:00:00 | 2021-04-15 |
| player5 | Game 247: Simulator | 2020-01-24 00:00:00 | 2021-06-24 |
| player5 | Game 1643: Adventure | 2020-09-23 00:00:00 | 2020-10-23 |
| player5 | Game 3485: Adventure | 2020-07-09 00:00:00 | 2021-06-28 |
| player5 | Game 3703: Simulator | 2020-01-27 00:00:00 | 2021-02-12 |
| player6 | Game 2133: Quest | 2020-06-04 00:00:00 | 2021-11-15 |
| player6 | Game 3944: Quest | 2020-05-31 00:00:00 | 2021-01-07 |

```
#   Time      Action                                                                                                                                      Message
13  19:40:29  SELECT d.name dev, round(avg(g.price)) avg_price, round(avg(g.reviewrating)) avg_Rating, count(g.gameid) game_count  FROM developers d JOIN g...   50 row(s) returned
14  19:40:37  EXPLAIN FORMAT=JSON SELECT p.username Username, g.title Game, pg.hoursplayed Hours_Registered FROM players_has_games pg JOIN games...          1 row(s) returned
15  19:40:45  SELECT p.username Username, g.title Game, pg.hoursplayed Hours_Registered FROM players_has_games pg JOIN games g ON pg.games_gameid = ...     3735 row(s) returned
16  19:41:38  SELECT d.name dev, round(avg(g.price)) avg_price, round(avg(g.reviewrating)) avg_Rating, count(g.gameid) game_count  FROM developers d JOIN g...   50 row(s) returned
17  19:42:05  SELECT d.name developer, p.name publisher, ROUND(AVG(g.price), 2) avg_price, ROUND(AVG(g.reviewrating), 1) avg_rating, COUNT(g.gameid) ga...     50 row(s) returned
18  19:42:49  SELECT p.username, g.title, pg.last_played, pg.purchase_date FROM players_has_games pg JOIN games g ON pg.games_gameid = g.gameid JOIN pl...    3683 row(s) returned
```

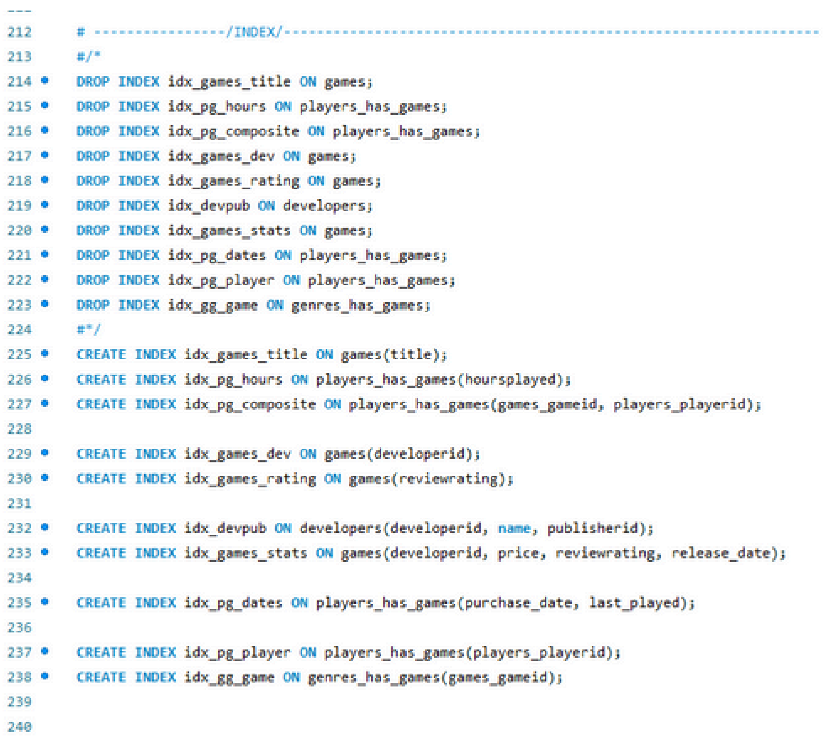


```sql
#Consulta 5 - Quantas horas o jogador possui em cada genero de jogo
SELECT p.username, gen.name favorite_genre, SUM(pg.hoursplayed) total_hours
FROM players p
JOIN players_has_games pg ON p.playerid = pg.players_playerid
JOIN games g ON pg.games_gameid = g.gameid
JOIN genres_has_games gg ON g.gameid = gg.games_gameid
JOIN genres gen ON gg.genres_genreid = gen.genreid
GROUP BY p.playerid, gen.name
ORDER BY p.username, total_hours DESC;
```

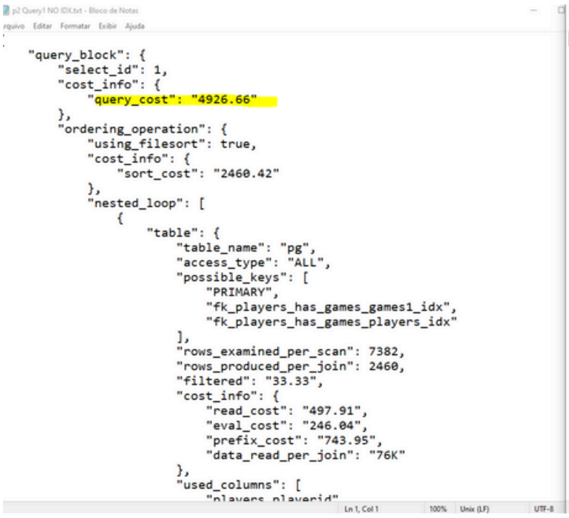
| username | favorite_genre | total_hours |
|---|---|---|
| player1 | Sports | 129.00 |
| player1 | FPS | 124.00 |
| player10 | RPG | 160.00 |
| player10 | Action | 67.00 |
| player10 | Adventure | 56.00 |
| player100 | Sports | 87.00 |
| player1000 | Adventure | 150.00 |
| player1000 | RPG | 84.00 |
| player1001 | FPS | 50.00 |
| player1001 | Action | 13.00 |
| player1002 | Action | 183.00 |
| player1003 | RPG | 39.00 |

```
#   Time      Action                                                                                                                                      Message
14  19:40:37  EXPLAIN FORMAT=JSON SELECT p.username Username, g.title Game, pg.hoursplayed Hours_Registered FROM players_has_games pg JOIN games...          1 row(s) returned
15  19:40:45  SELECT p.username Username, g.title Game, pg.hoursplayed Hours_Registered FROM players_has_games pg JOIN games g ON pg.games_gameid = ...     3735 row(s) returned
16  19:41:38  SELECT d.name dev, round(avg(g.price)) avg_price, round(avg(g.reviewrating)) avg_Rating, count(g.gameid) game_count  FROM developers d JOIN g...   50 row(s) returned
17  19:42:05  SELECT d.name developer, p.name publisher, ROUND(AVG(g.price), 2) avg_price, ROUND(AVG(g.reviewrating), 1) avg_rating, COUNT(g.gameid) ga...     50 row(s) returned
18  19:42:49  SELECT p.username, g.title, pg.last_played, pg.purchase_date FROM players_has_games pg JOIN games g ON pg.games_gameid = g.gameid JOIN pl...    3683 row(s) returned
19  19:43:14  SELECT p.username, gen.name favorite_genre, SUM(pg.hoursplayed) total_hours FROM players p JOIN players_has_games pg ON p.playerid = pg.pla...   5708 row(s) returned
```

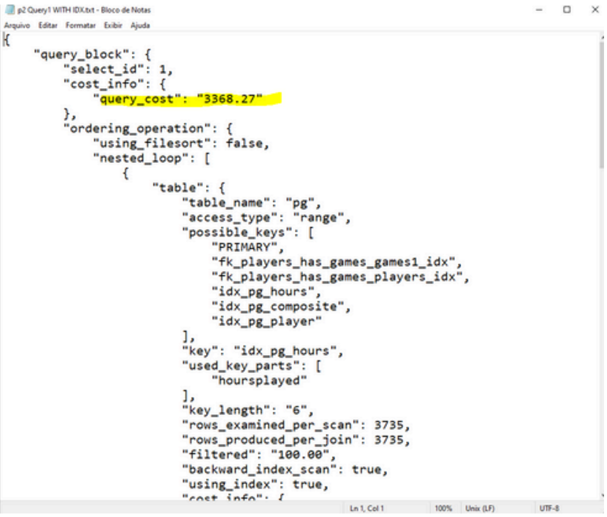
# 4. Otimizar



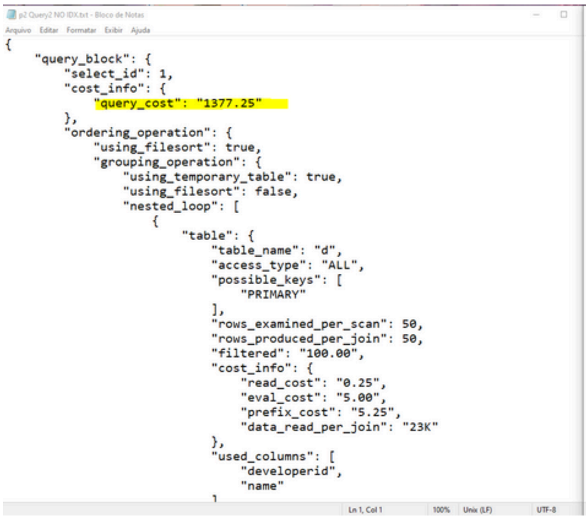
```sql
# ---------------/INDEX/-----------------------------------------------------------------
#/*
DROP INDEX idx_games_title ON games;
DROP INDEX idx_pg_hours ON players_has_games;
DROP INDEX idx_pg_composite ON players_has_games;
DROP INDEX idx_games_dev ON games;
DROP INDEX idx_games_rating ON games;
DROP INDEX idx_devpub ON developers;
DROP INDEX idx_games_stats ON games;
DROP INDEX idx_pg_dates ON players_has_games;
DROP INDEX idx_pg_player ON players_has_games;
DROP INDEX idx_gg_game ON genres_has_games;
#*/
CREATE INDEX idx_games_title ON games(title);
CREATE INDEX idx_pg_hours ON players_has_games(hoursplayed);
CREATE INDEX idx_pg_composite ON players_has_games(games_gameid, players_playerid);

CREATE INDEX idx_games_dev ON games(developerid);
CREATE INDEX idx_games_rating ON games(reviewrating);

CREATE INDEX idx_devpub ON developers(developerid, name, publisherid);
CREATE INDEX idx_games_stats ON games(developerid, price, reviewrating, release_date);

CREATE INDEX idx_pg_dates ON players_has_games(purchase_date, last_played);

CREATE INDEX idx_pg_player ON players_has_games(players_playerid);
CREATE INDEX idx_gg_game ON genres_has_games(games_gameid);
```


p2 Query1 NO IDX.txt - Bloco de Notas
rquivo Editar Formatar Exibir Ajuda
Ln 1, Col 1
100%
Unix (LF)
UTF-8

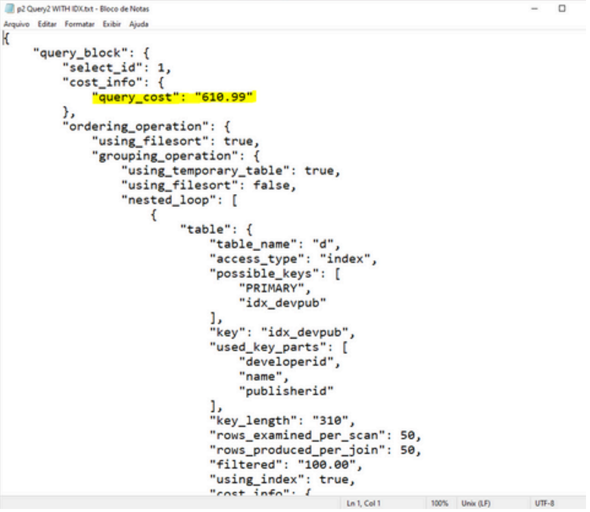
```json
    "query_block": {
        "select_id": 1,
        "cost_info": {
            "query_cost": "4926.66"
        },
        "ordering_operation": {
            "using_filesort": true,
            "cost_info": {
                "sort_cost": "2460.42"
            },
            "nested_loop": [
                {
                    "table": {
                        "table_name": "pg",
                        "access_type": "ALL",
                        "possible_keys": [
                            "PRIMARY",
                            "fk_players_has_games_games1_idx",
                            "fk_players_has_games_players_idx"
                        ],
                        "rows_examined_per_scan": 7382,
                        "rows_produced_per_join": 2460,
                        "filtered": "33.33",
                        "cost_info": {
                            "read_cost": "497.91",
                            "eval_cost": "246.04",
                            "prefix_cost": "743.95",
                            "data_read_per_join": "76K"
                        },
                        "used_columns": [
                            "players_playerid"
```


p2 Query1 WITH IDX.txt - Bloco de Notas
Arquivo Editar Formatar Exibir Ajuda
Ln 1, Col 1
100%
Unix (LF)
UTF-8

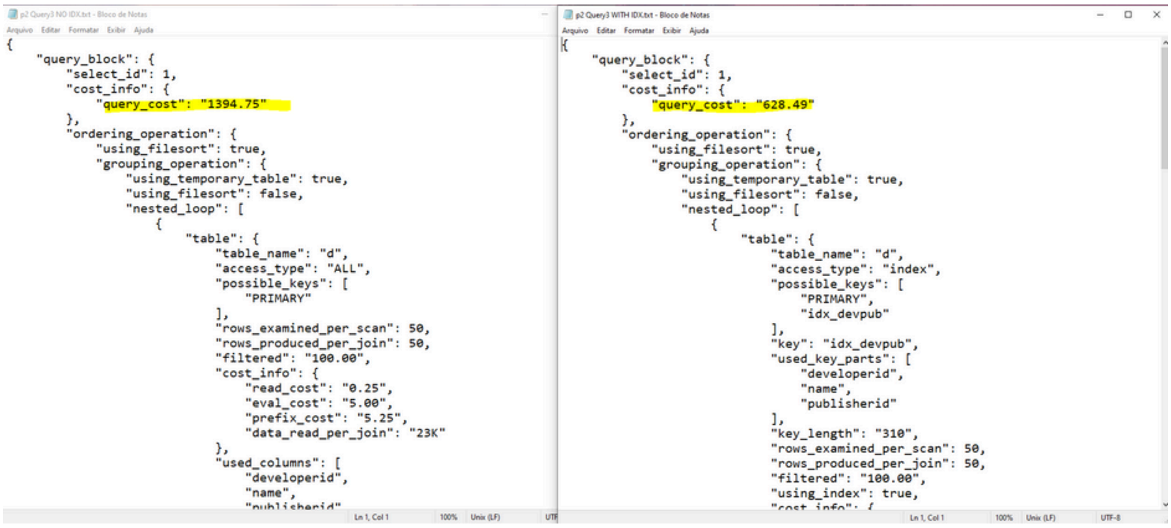
```json
{
    "query_block": {
        "select_id": 1,
        "cost_info": {
            "query_cost": "3368.27"
        },
        "ordering_operation": {
            "using_filesort": false,
            "nested_loop": [
                {
                    "table": {
                        "table_name": "pg",
                        "access_type": "range",
                        "possible_keys": [
                            "PRIMARY",
                            "fk_players_has_games_games1_idx",
                            "fk_players_has_games_players_idx",
                            "idx_pg_hours",
                            "idx_pg_composite",
                            "idx_pg_player"
                        ],
                        "key": "idx_pg_hours",
                        "used_key_parts": [
                            "hoursplayed"
                        ],
                        "key_length": "6",
                        "rows_examined_per_scan": 3735,
                        "rows_produced_per_join": 3735,
                        "filtered": "100.00",
                        "backward_index_scan": true,
                        "using_index": true,
                        "cost_info": {
```


p2 Query2 NO IDX.txt - Bloco de Notas
Arquivo Editar Formatar Exibir Ajuda
Ln 1, Col 1
100%
Unix (LF)
UTF-8

```json
{
    "query_block": {
        "select_id": 1,
        "cost_info": {
            "query_cost": "1377.25"
        },
        "ordering_operation": {
            "using_filesort": true,
            "grouping_operation": {
                "using_temporary_table": true,
                "using_filesort": false,
                "nested_loop": [
                    {
                        "table": {
                            "table_name": "d",
                            "access_type": "ALL",
                            "possible_keys": [
                                "PRIMARY"
                            ],
                            "rows_examined_per_scan": 50,
                            "rows_produced_per_join": 50,
                            "filtered": "100.00",
                            "cost_info": {
                                "read_cost": "0.25",
                                "eval_cost": "5.00",
                                "prefix_cost": "5.25",
                                "data_read_per_join": "23K"
                            },
                            "used_columns": [
                                "developerid",
                                "name"
                            ]
```


p2 Query2 WITH IDX.txt - Bloco de Notas
Arquivo Editar Formatar Exibir Ajuda
Ln 1, Col 1
100%
Unix (LF)
UTF-8

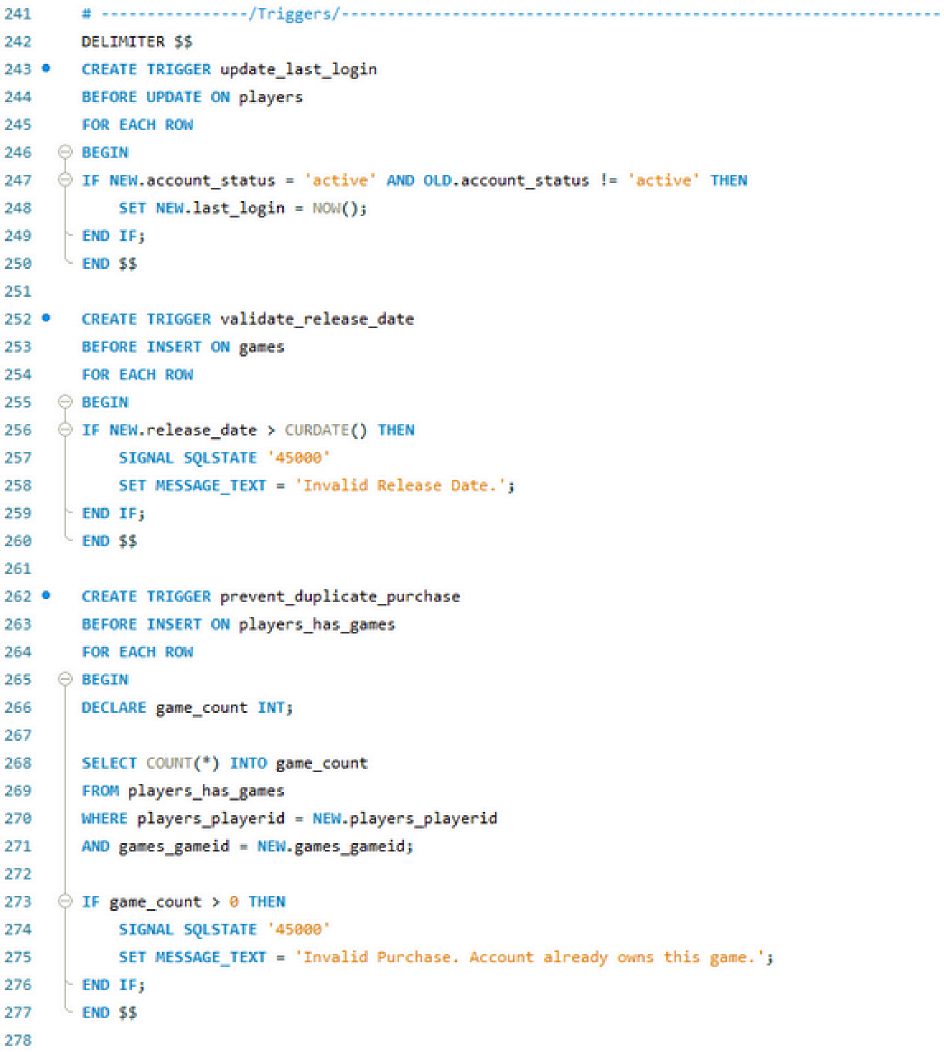
```json
{
    "query_block": {
        "select_id": 1,
        "cost_info": {
            "query_cost": "610.99"
        },
        "ordering_operation": {
            "using_filesort": true,
            "grouping_operation": {
                "using_temporary_table": true,
                "using_filesort": false,
                "nested_loop": [
                    {
                        "table": {
                            "table_name": "d",
                            "access_type": "index",
                            "possible_keys": [
                                "PRIMARY",
                                "idx_devpub"
                            ],
                            "key": "idx_devpub",
                            "used_key_parts": [
                                "developerid",
                                "name",
                                "publisherid"
                            ],
                            "key_length": "310",
                            "rows_examined_per_scan": 50,
                            "rows_produced_per_join": 50,
                            "filtered": "100.00",
                            "using_index": true,
                            "cost_info": {
```


p2 Query3 NO IDX.txt - Bloco de Notas
Arquivo Editar Formatar Exibir Ajuda
Ln 1, Col 1
100%
Unix (LF)
UTF
p2 Query3 WITH IDX.txt - Bloco de Notas
Arquivo Editar Formatar Exibir Ajuda
Ln 1, Col 1
100%
Unix (LF)
UTF-8

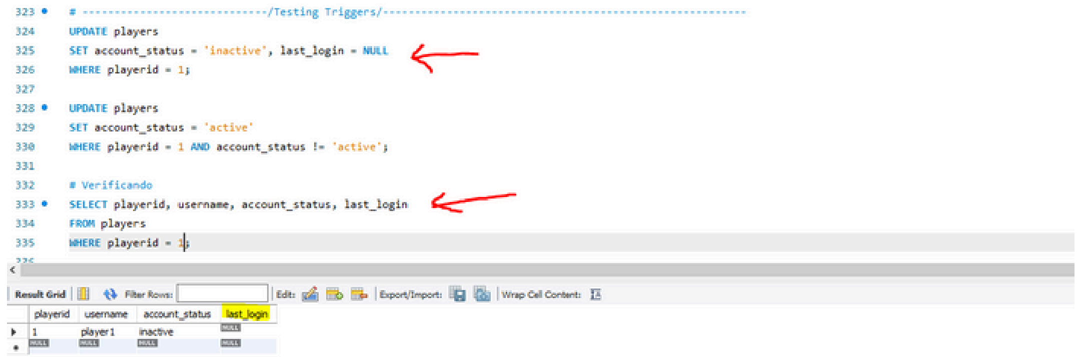
```
{
    "query_block": {
        "select_id": 1,
        "cost_info": {
            "query_cost": "1394.75"
        },
        "ordering_operation": {
            "using_filesort": true,
            "grouping_operation": {
                "using_temporary_table": true,
                "using_filesort": false,
                "nested_loop": [
                    {
                        "table": {
                            "table_name": "d",
                            "access_type": "ALL",
                            "possible_keys": [
                                "PRIMARY"
                            ],
                            "rows_examined_per_scan": 50,
                            "rows_produced_per_join": 50,
                            "filtered": "100.00",
                            "cost_info": {
                                "read_cost": "0.25",
                                "eval_cost": "5.00",
                                "prefix_cost": "5.25",
                                "data_read_per_join": "23K"
                            },
                            "used_columns": [
                                "developerid",
                                "name",
                                "publisherid"
```

```
{
    "query_block": {
        "select_id": 1,
        "cost_info": {
            "query_cost": "628.49"
        },
        "ordering_operation": {
            "using_filesort": true,
            "grouping_operation": {
                "using_temporary_table": true,
                "using_filesort": false,
                "nested_loop": [
                    {
                        "table": {
                            "table_name": "d",
                            "access_type": "index",
                            "possible_keys": [
                                "PRIMARY",
                                "idx_devpub"
                            ],
                            "key": "idx_devpub",
                            "used_key_parts": [
                                "developerid",
                                "name",
                                "publisherid"
                            ],
                            "key_length": "310",
                            "rows_examined_per_scan": 50,
                            "rows_produced_per_join": 50,
                            "filtered": "100.00",
                            "using_index": true,
                            "cost_info": {
```

# 5.1 Estruturas avançadas (Triggers)

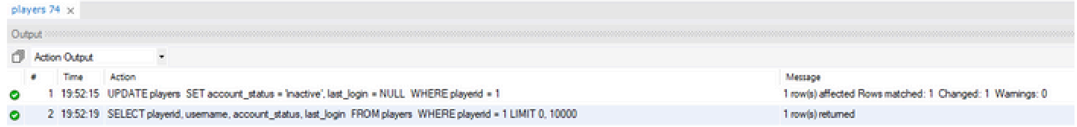


```sql
# ---------------/Triggers/-----------------------------------------------------------------
DELIMITER $$
CREATE TRIGGER update_last_login
BEFORE UPDATE ON players
FOR EACH ROW
BEGIN
IF NEW.account_status = 'active' AND OLD.account_status != 'active' THEN
    SET NEW.last_login = NOW();
END IF;
END $$

CREATE TRIGGER validate_release_date
BEFORE INSERT ON games
FOR EACH ROW
BEGIN
IF NEW.release_date > CURDATE() THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Invalid Release Date.';
END IF;
END $$

CREATE TRIGGER prevent_duplicate_purchase
BEFORE INSERT ON players_has_games
FOR EACH ROW
BEGIN
DECLARE game_count INT;

SELECT COUNT(*) INTO game_count
FROM players_has_games
WHERE players_playerid = NEW.players_playerid
AND games_gameid = NEW.games_gameid;

IF game_count > 0 THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Invalid Purchase. Account already owns this game.';
END IF;
END $$

```

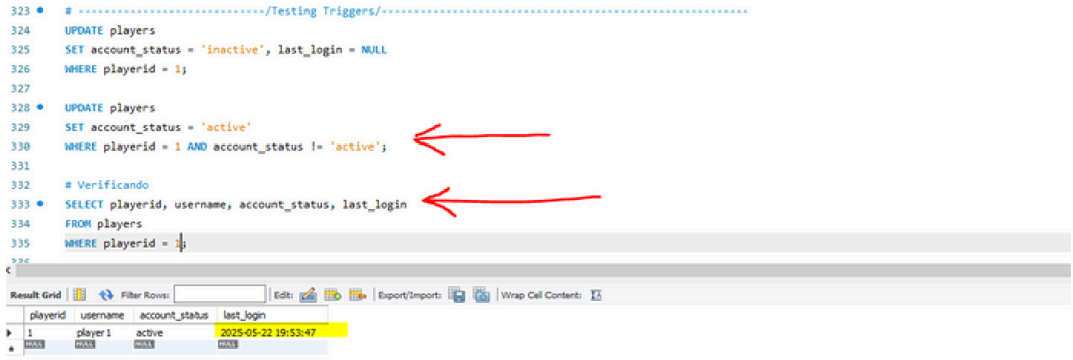


```
# ---------------------------/Testing Triggers/-------------------------------------------------------
UPDATE players
SET account_status = 'inactive', last_login = NULL
WHERE playerid = 1;

UPDATE players
SET account_status = 'active'
WHERE playerid = 1 AND account_status != 'active';

# Verificando
SELECT playerid, username, account_status, last_login
FROM players
WHERE playerid = 1;
```

```
playerid  username  account_status  last_login
1         player1   inactive        NULL
NULL      NULL      NULL            NULL
```

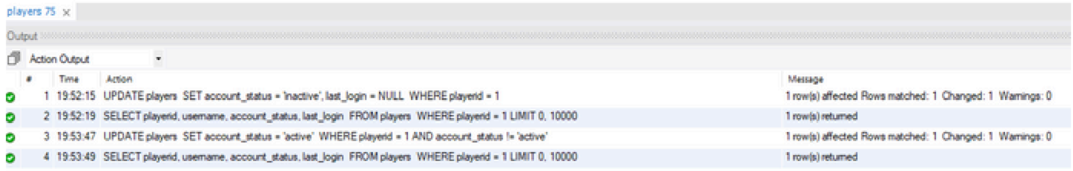


```
#  Time      Action                                                                              Message
1  19:52:15  UPDATE players  SET account_status = 'inactive', last_login = NULL  WHERE playerid = 1  1 row(s) affected Rows matched: 1  Changed: 1  Warnings: 0
2  19:52:19  SELECT playerid, username, account_status, last_login  FROM players  WHERE playerid = 1 LIMIT 0, 10000  1 row(s) returned
```

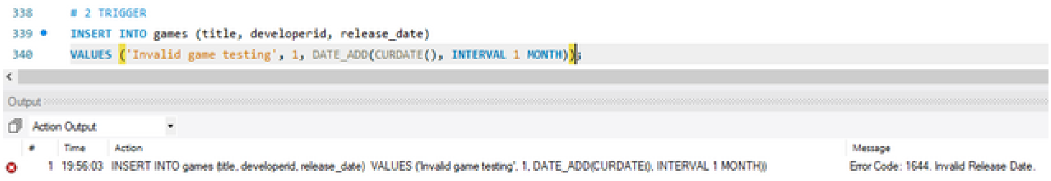


```
# ---------------------------/Testing Triggers/-------------------------------------------------------
UPDATE players
SET account_status = 'inactive', last_login = NULL
WHERE playerid = 1;

UPDATE players
SET account_status = 'active'
WHERE playerid = 1 AND account_status != 'active';

# Verificando
SELECT playerid, username, account_status, last_login
FROM players
WHERE playerid = 1;
```

```
playerid  username  account_status  last_login
1         player1   active          2025-05-22 19:53:47
NULL      NULL      NULL            NULL
```

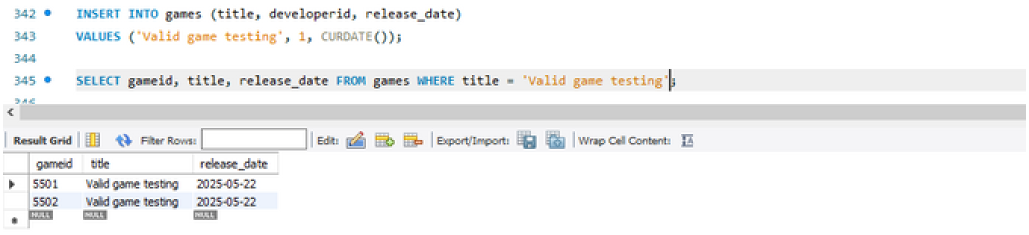


```
#  Time      Action                                                                              Message
1  19:52:15  UPDATE players  SET account_status = 'inactive', last_login = NULL  WHERE playerid = 1  1 row(s) affected Rows matched: 1  Changed: 1  Warnings: 0
2  19:52:19  SELECT playerid, username, account_status, last_login  FROM players  WHERE playerid = 1 LIMIT 0, 10000  1 row(s) returned
3  19:53:47  UPDATE players  SET account_status = 'active'  WHERE playerid = 1 AND account_status != 'active'  1 row(s) affected Rows matched: 1  Changed: 1  Warnings: 0
4  19:53:49  SELECT playerid, username, account_status, last_login  FROM players  WHERE playerid = 1 LIMIT 0, 10000  1 row(s) returned
```



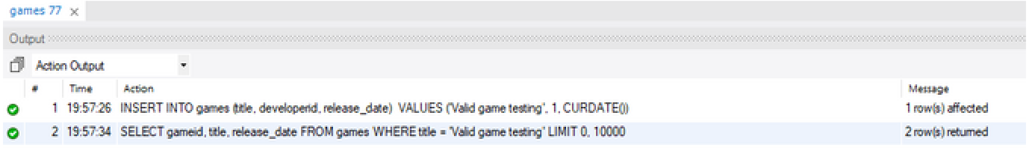
```
# 2 TRIGGER
INSERT INTO games (title, developerid, release_date)
VALUES ('Invalid game testing', 1, DATE_ADD(CURDATE(), INTERVAL 1 MONTH));
```

```
#  Time      Action                                                                                                      Message
1  19:56:03  INSERT INTO games (title, developerid, release_date)  VALUES ('Invalid game testing', 1, DATE_ADD(CURDATE(), INTERVAL 1 MONTH))  Error Code: 1644. Invalid Release Date.
```

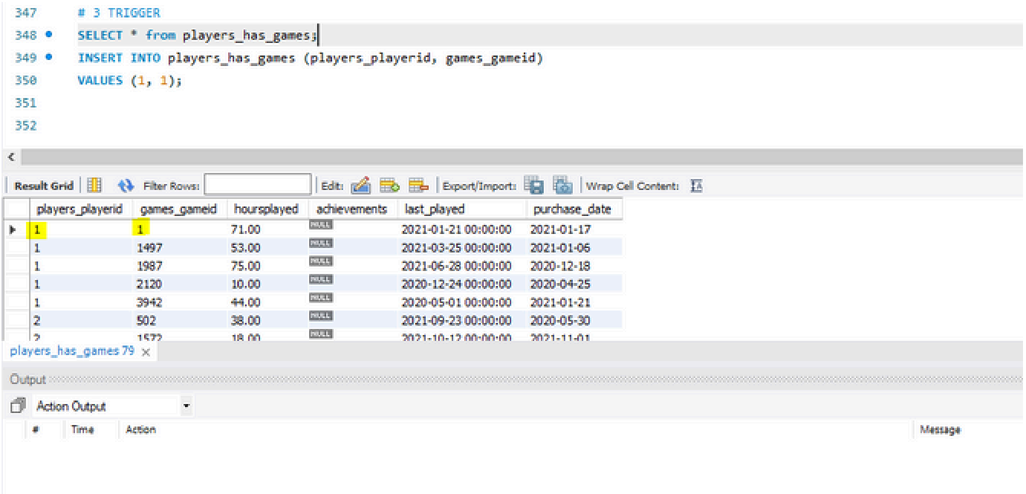


```
INSERT INTO games (title, developerid, release_date)
VALUES ('Valid game testing', 1, CURDATE());

SELECT gameid, title, release_date FROM games WHERE title = 'Valid game testing';
```

```
gameid  title               release_date
5501    Valid game testing  2025-05-22
5502    Valid game testing  2025-05-22
NULL    NULL                NULL
```

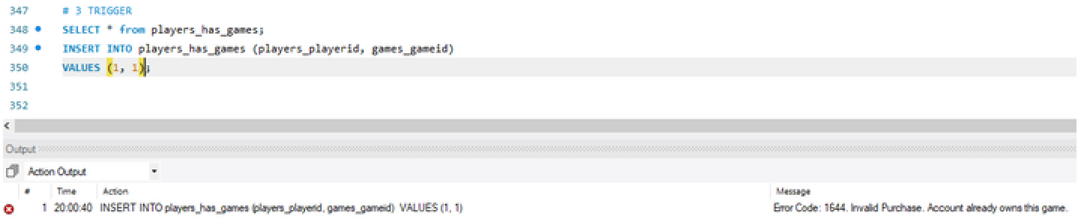


```
#  Time      Action                                                                                     Message
1  19:57:26  INSERT INTO games (title, developerid, release_date)  VALUES ('Valid game testing', 1, CURDATE())  1 row(s) affected
2  19:57:34  SELECT gameid, title, release_date FROM games WHERE title = 'Valid game testing' LIMIT 0, 10000  2 row(s) returned
```



```
# 3 TRIGGER
SELECT * from players_has_games;
INSERT INTO players_has_games (players_playerid, games_gameid)
VALUES (1, 1);
```

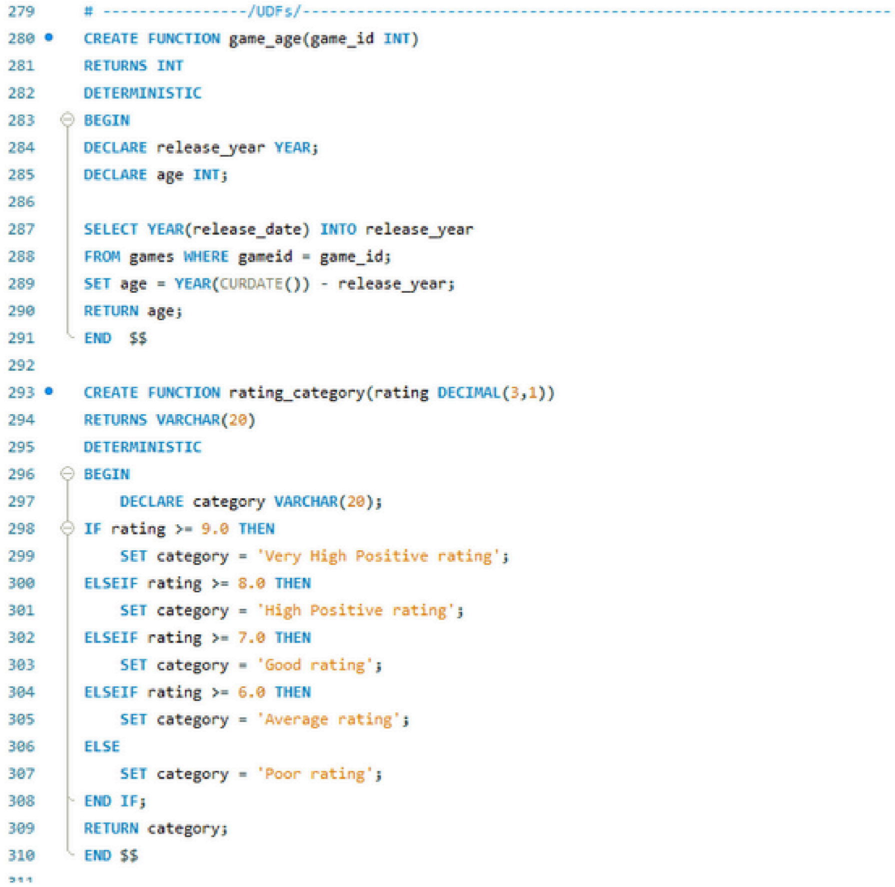
| players_playerid | games_gameid | hoursplayed | achievements | last_played | purchase_date |
|---|---|---|---|---|---|
| 1 | 1 | 71.00 | NULL | 2021-01-21 00:00:00 | 2021-01-17 |
| 1 | 1497 | 53.00 | NULL | 2021-03-25 00:00:00 | 2021-01-06 |
| 1 | 1987 | 75.00 | NULL | 2021-06-28 00:00:00 | 2020-12-18 |
| 1 | 2120 | 10.00 | NULL | 2020-12-24 00:00:00 | 2020-04-25 |
| 1 | 3942 | 44.00 | NULL | 2020-05-01 00:00:00 | 2021-01-21 |
| 2 | 502 | 38.00 | NULL | 2021-09-23 00:00:00 | 2020-05-30 |
| 2 | 1572 | 18.00 | NULL | 2021-10-12 00:00:00 | 2021-11-01 |



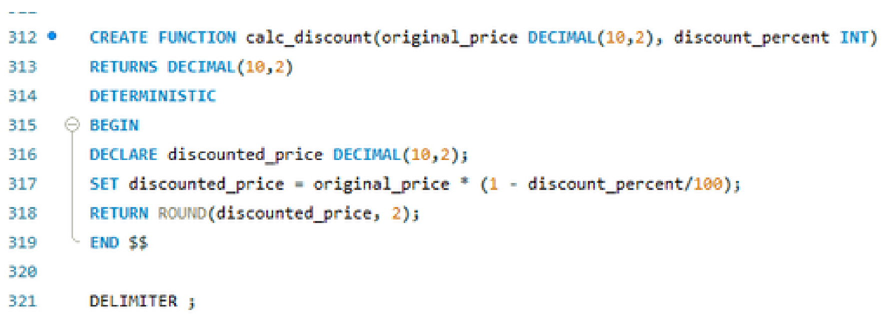
```
# 3 TRIGGER
SELECT * from players_has_games;
INSERT INTO players_has_games (players_playerid, games_gameid)
VALUES (1, 1);
```

```
#  Time      Action                                                                   Message
1  20:00:40  INSERT INTO players_has_games (players_playerid, games_gameid)  VALUES (1, 1)   Error Code: 1644. Invalid Purchase. Account already owns this game.
```

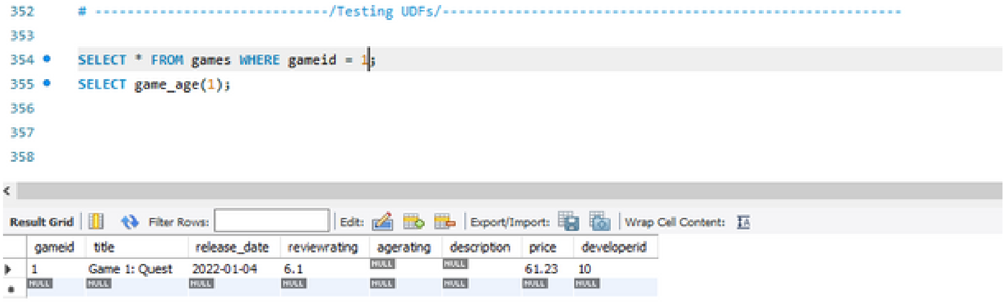
# 5.2 Estruturas avançadas (UDFs)



```
# ----------------/UDFs/-------------------------------------------------------------------
CREATE FUNCTION game_age(game_id INT)
RETURNS INT
DETERMINISTIC
BEGIN
DECLARE release_year YEAR;
DECLARE age INT;

SELECT YEAR(release_date) INTO release_year
FROM games WHERE gameid = game_id;
SET age = YEAR(CURDATE()) - release_year;
RETURN age;
END  $$

CREATE FUNCTION rating_category(rating DECIMAL(3,1))
RETURNS VARCHAR(20)
DETERMINISTIC
BEGIN
    DECLARE category VARCHAR(20);
IF rating >= 9.0 THEN
    SET category = 'Very High Positive rating';
ELSEIF rating >= 8.0 THEN
    SET category = 'High Positive rating';
ELSEIF rating >= 7.0 THEN
    SET category = 'Good rating';
ELSEIF rating >= 6.0 THEN
    SET category = 'Average rating';
ELSE
    SET category = 'Poor rating';
END IF;
RETURN category;
END $$
```

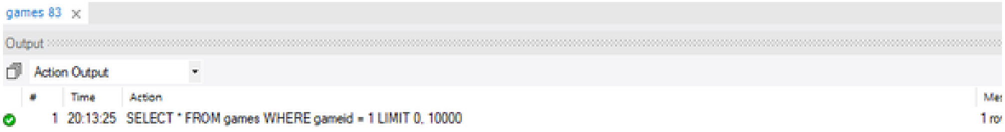


```
CREATE FUNCTION calc_discount(original_price DECIMAL(10,2), discount_percent INT)
RETURNS DECIMAL(10,2)
DETERMINISTIC
BEGIN
DECLARE discounted_price DECIMAL(10,2);
SET discounted_price = original_price * (1 - discount_percent/100);
RETURN ROUND(discounted_price, 2);
END $$

DELIMITER ;
```

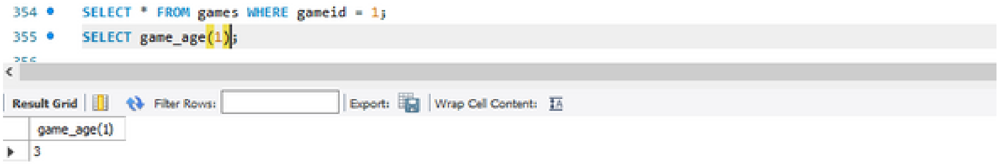


```
# ----------------------------/Testing UDFs/-----------------------------------------------------------

SELECT * FROM games WHERE gameid = 1;
SELECT game_age(1);
```

| gameid | title | release_date | reviewrating | agerating | description | price | developerid |
|---|---|---|---|---|---|---|---|
| 1 | Game 1: Quest | 2022-01-04 | 6.1 | NULL | NULL | 61.23 | 10 |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL | NULL |

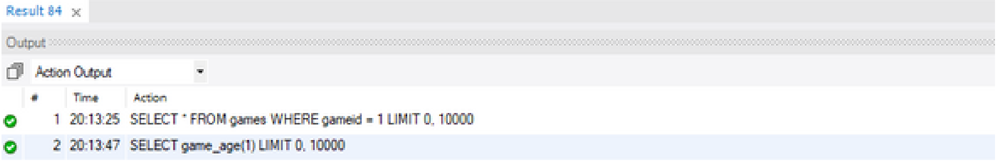


```
#   Time      Action                                                Me
1   20:13:25  SELECT * FROM games WHERE gameid = 1 LIMIT 0, 10000   1 ro
```

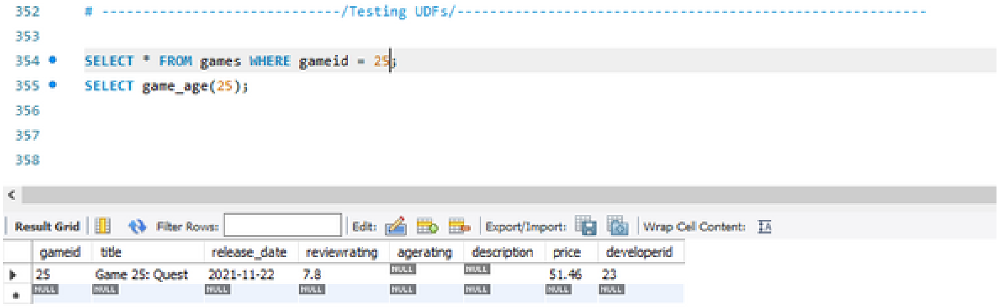


```
SELECT * FROM games WHERE gameid = 1;
SELECT game_age(1);
```

| game_age(1) |
|---|
| 3 |

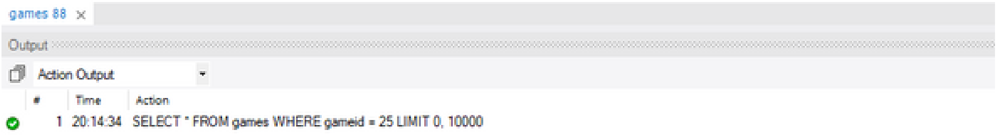


```
#   Time      Action
1   20:13:25  SELECT * FROM games WHERE gameid = 1 LIMIT 0, 10000
2   20:13:47  SELECT game_age(1) LIMIT 0, 10000
```

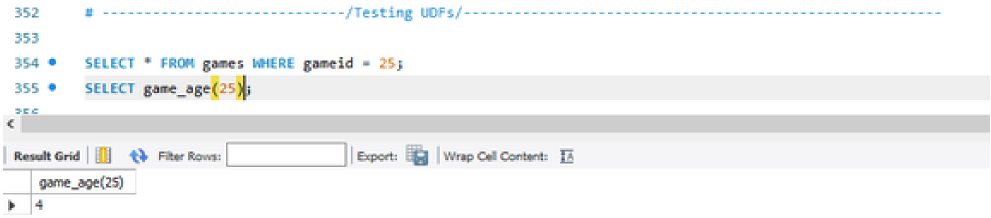


```
# ----------------------------/Testing UDFs/-----------------------------------------------------------

SELECT * FROM games WHERE gameid = 25;
SELECT game_age(25);
```

| gameid | title | release_date | reviewrating | agerating | description | price | developerid |
|---|---|---|---|---|---|---|---|
| 25 | Game 25: Quest | 2021-11-22 | 7.8 | NULL | NULL | 51.46 | 23 |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL | NULL |

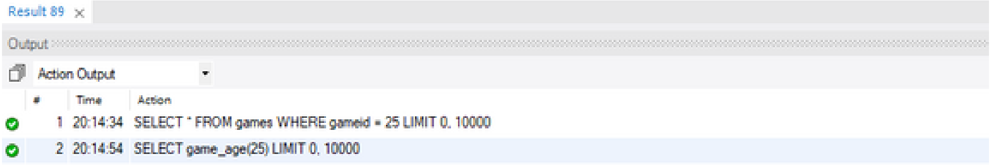


```
#   Time      Action
1   20:14:34  SELECT * FROM games WHERE gameid = 25 LIMIT 0, 10000
```

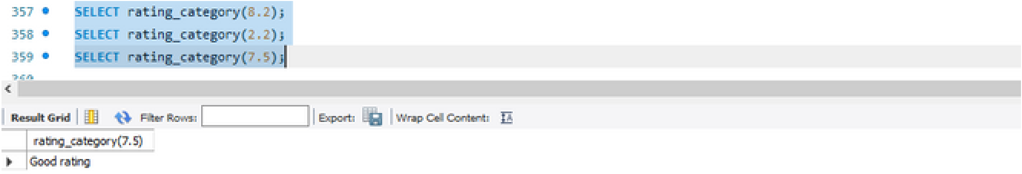


```
# ----------------------------/Testing UDFs/--------------------------------------------------------

SELECT * FROM games WHERE gameid = 25;
SELECT game_age(25);
```

```
game_age(25)
4
```

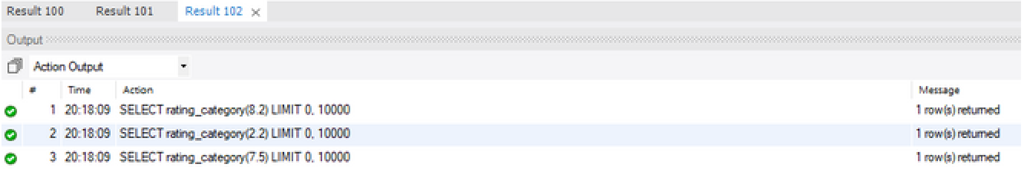


```
   #  Time      Action
   1  20:14:34  SELECT * FROM games WHERE gameid = 25 LIMIT 0, 10000
   2  20:14:54  SELECT game_age(25) LIMIT 0, 10000
```

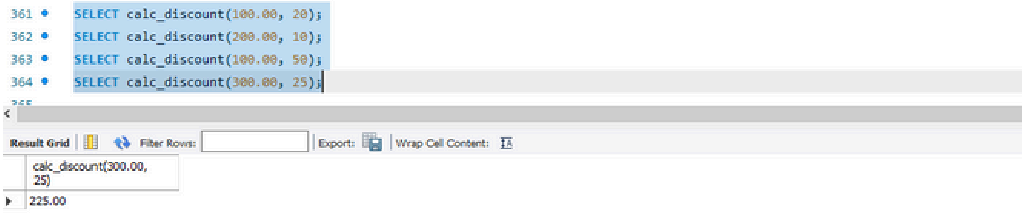


```
SELECT rating_category(8.2);
SELECT rating_category(2.2);
SELECT rating_category(7.5);
```

```
rating_category(7.5)
Good rating
```

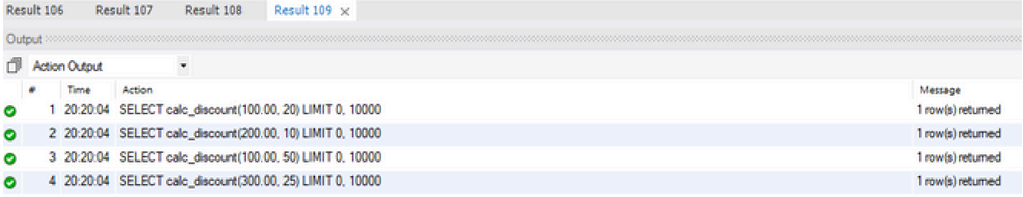


```
   #  Time      Action                                        Message
   1  20:18:09  SELECT rating_category(8.2) LIMIT 0, 10000    1 row(s) returned
   2  20:18:09  SELECT rating_category(2.2) LIMIT 0, 10000    1 row(s) returned
   3  20:18:09  SELECT rating_category(7.5) LIMIT 0, 10000    1 row(s) returned
```



```
SELECT calc_discount(100.00, 20);
SELECT calc_discount(200.00, 10);
SELECT calc_discount(100.00, 50);
SELECT calc_discount(300.00, 25);
```

```
calc_discount(300.00,
25)
225.00
```



```
   #  Time      Action                                               Message
   1  20:20:04  SELECT calc_discount(100.00, 20) LIMIT 0, 10000      1 row(s) returned
   2  20:20:04  SELECT calc_discount(200.00, 10) LIMIT 0, 10000      1 row(s) returned
   3  20:20:04  SELECT calc_discount(100.00, 50) LIMIT 0, 10000      1 row(s) returned
   4  20:20:04  SELECT calc_discount(300.00, 25) LIMIT 0, 10000      1 row(s) returned
```