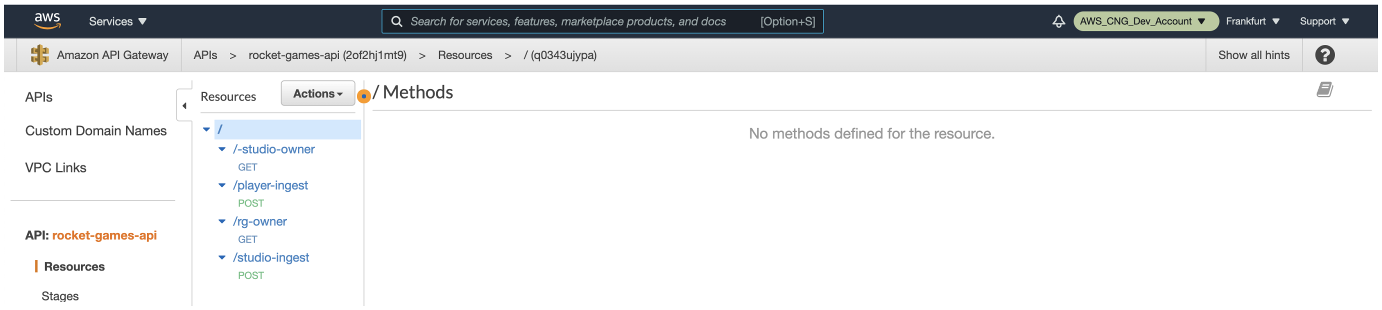
User Story Results

User story1:

As a player, I want to create an account and register to play multiple games.

Assumed, these details will come from UI and kick the API for feeding into backend DB. DB designed according to this model. Below is the API snapshot.

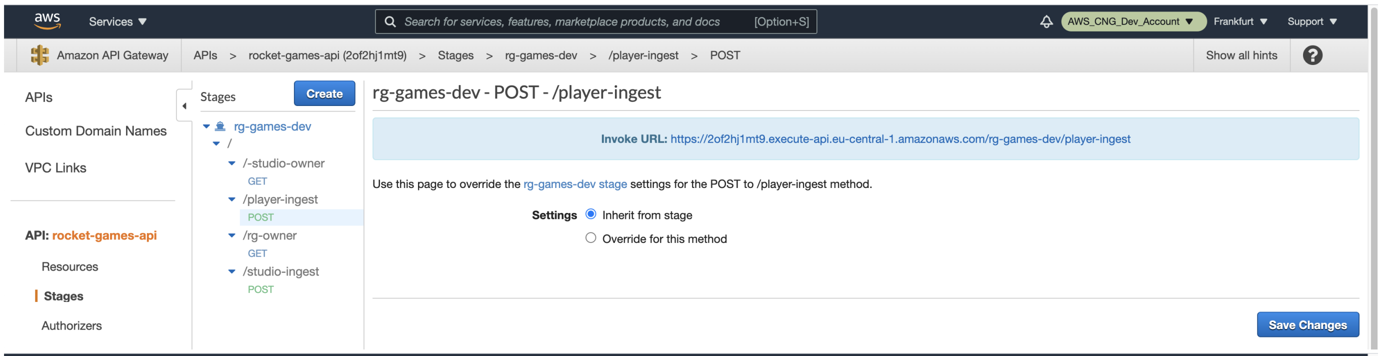


Resources

- /
  - /-studio-owner GET
  - /player-ingest POST
  - /rg-owner GET
  - /studio-ingest POST

/ Methods

No methods defined for the resource.



Stages

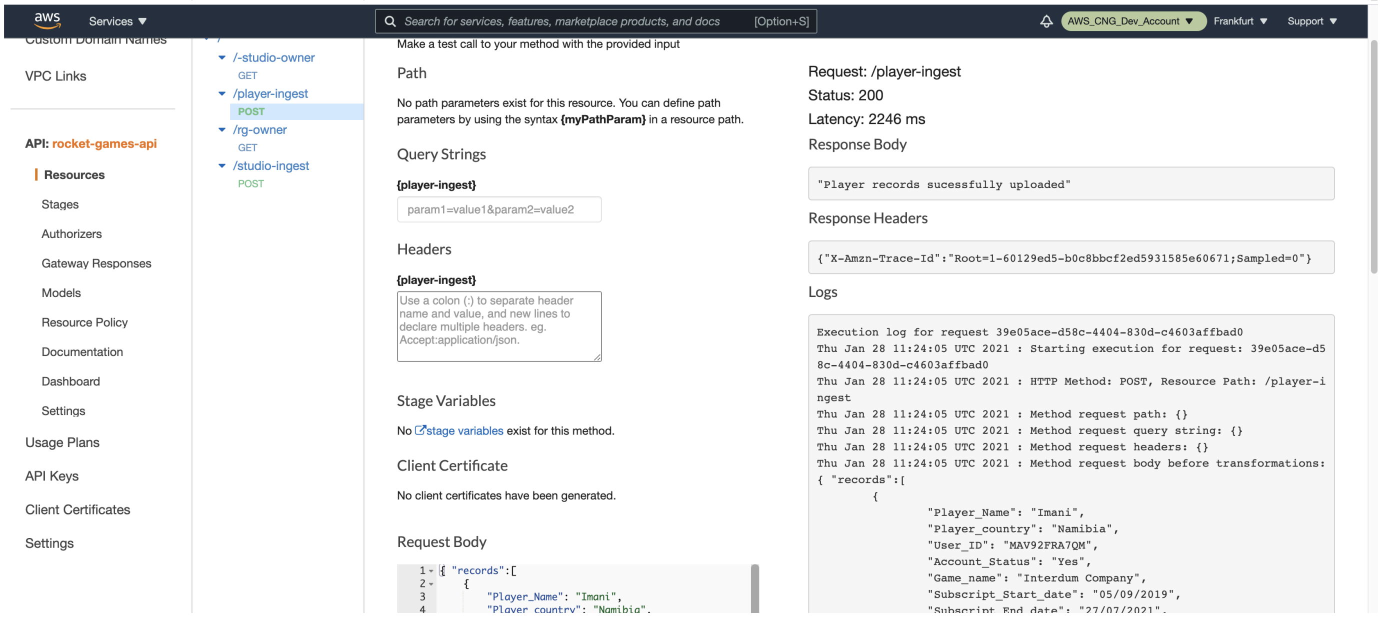
- rg-games-dev
  - /
    - /-studio-owner GET
    - /player-ingest POST
    - /rg-owner GET
    - /studio-ingest POST

rg-games-dev - POST - /player-ingest

Invoke URL: https://2of2hj1mt9.execute-api.eu-central-1.amazonaws.com/rg-games-dev/player-ingest

Use this page to override the rg-games-dev stage settings for the POST to /player-ingest method.

Settings (●) Inherit from stage ( ) Override for this method



- /-studio-owner GET
- /player-ingest POST
- /rg-owner GET
- /studio-ingest POST

Make a test call to your method with the provided input

Path

No path parameters exist for this resource. You can define path parameters by using the syntax **{myPathParam}** in a resource path.

Query Strings

{player-ingest}

param1=value1&param2=value2

Headers

{player-ingest}

Use a colon (:) to separate header name and value, and new lines to declare multiple headers. eg. Accept:application/json.

Stage Variables

No stage variables exist for this method.

Client Certificate

No client certificates have been generated.

Request Body

```
{ "records":[
    {
        "Player_Name": "Imani",
        "Player_country": "Namibia",
```

Request: /player-ingest
Status: 200
Latency: 2246 ms

Response Body

```
"Player records sucessfully uploaded"
```

Response Headers

```
{"X-Amzn-Trace-Id":"Root=1-60129ed5-b0c8bbcf2ed5931585e60671;Sampled=0"}
```

Logs

```
Execution log for request 39e05ace-d58c-4404-830d-c4603affbad0
Thu Jan 28 11:24:05 UTC 2021 : Starting execution for request: 39e05ace-d5
8c-4404-830d-c4603affbad0
Thu Jan 28 11:24:05 UTC 2021 : HTTP Method: POST, Resource Path: /player-i
ngest
Thu Jan 28 11:24:05 UTC 2021 : Method request path: {}
Thu Jan 28 11:24:05 UTC 2021 : Method request query string: {}
Thu Jan 28 11:24:05 UTC 2021 : Method request headers: {}
Thu Jan 28 11:24:05 UTC 2021 : Method request body before transformations:
{ "records":[
        {
                "Player_Name": "Imani",
                "Player_country": "Namibia",
                "User_ID": "MAV92FRA7QM",
                "Account_Status": "Yes",
                "Game_name": "Interdum Company",
                "Subscript_Start_date": "05/09/2019",
                "Subscript_End_date": "27/07/2021",
```

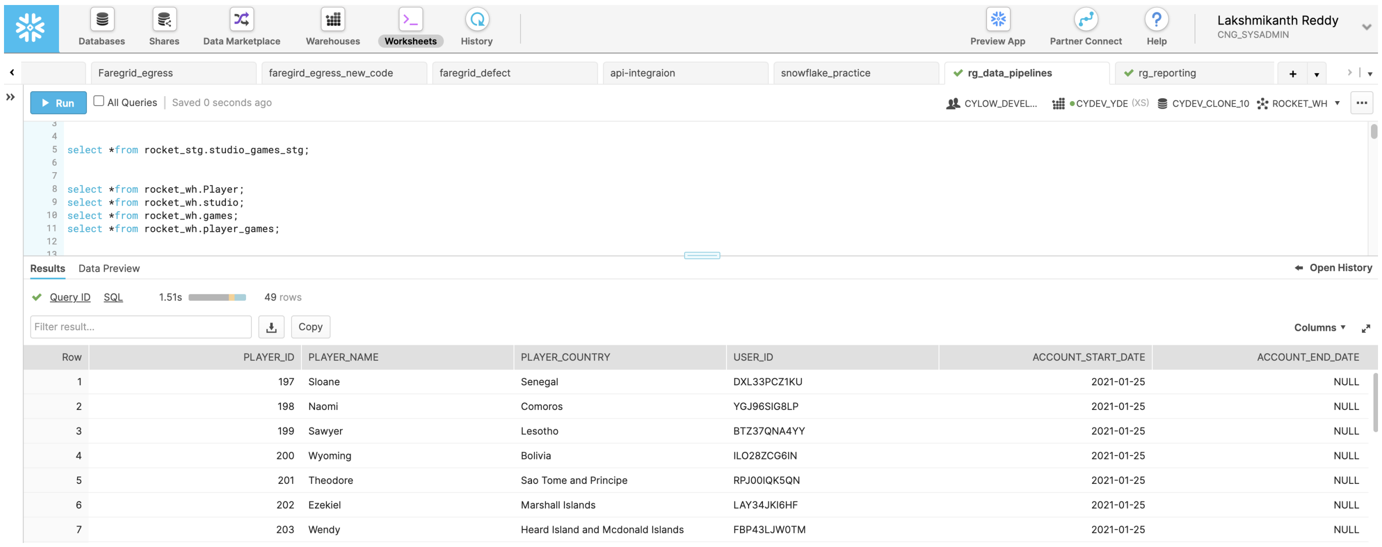
Note: Player records will be ingested to S3 bucket as per design.

User story2:

As a player, I want to be able to close my account

Assumed this record will as `"Account_Status": "N"` from through UI. This will be consumed as CDC record for that player and necessary update will happen at DB level by using data-pipelines. So associated record status will change as Account_End_Date as close request date else null for active records in Player table.

```
"Player_Name": "Imani",
   "Player_country": "Namibia",
   "User_ID": "MAV92FRA7QM",
   "Account_Status": "Yes",
   "Game_name": "Interdum Company",
   "Game_Subscript_Start_date": "05/09/2019",
   "Game_Subscription_End_date": "27/07/2021",
   "Account_Status": "N"
},
```



```
select *from rocket_stg.studio_games_stg;


select *from rocket_wh.Player;
select *from rocket_wh.studio;
select *from rocket_wh.games;
select *from rocket_wh.player_games;
```

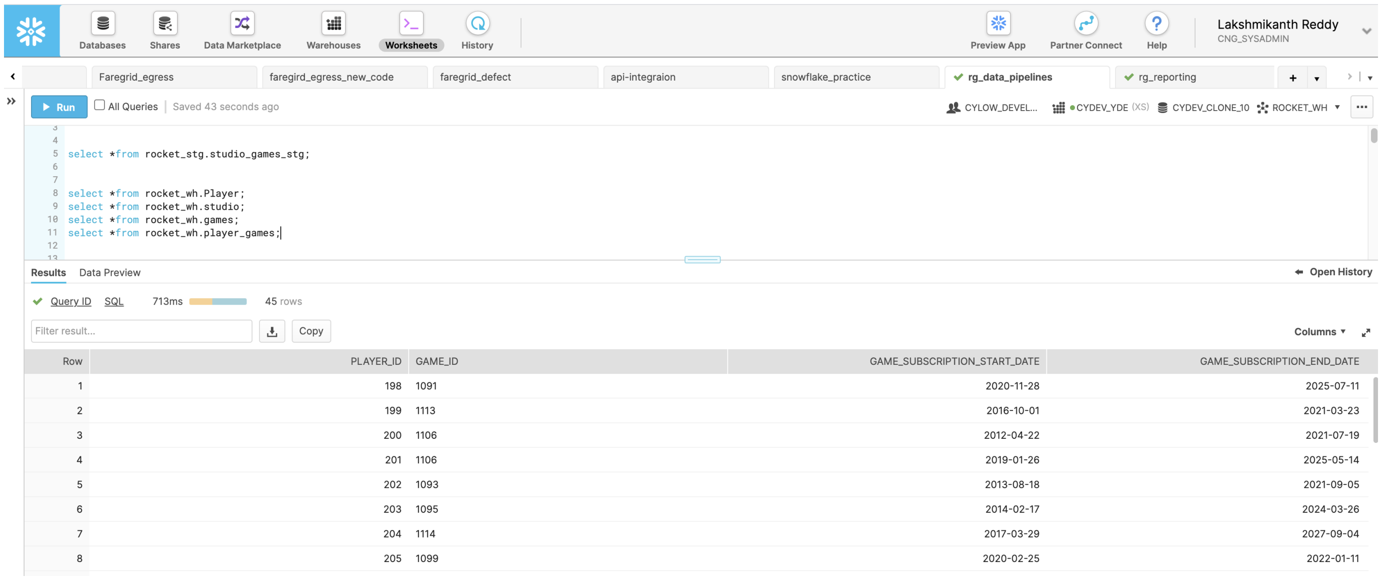
Query ID SQL 1.51s 49 rows

Row	PLAYER_ID	PLAYER_NAME	PLAYER_COUNTRY	USER_ID	ACCOUNT_START_DATE	ACCOUNT_END_DATE
1	197	Sloane	Senegal	DXL33PCZ1KU	2021-01-25	NULL
2	198	Naomi	Comoros	YGJ96SIG8LP	2021-01-25	NULL
3	199	Sawyer	Lesotho	BTZ37QNA4YY	2021-01-25	NULL
4	200	Wyoming	Bolivia	ILO28ZCG6IN	2021-01-25	NULL
5	201	Theodore	Sao Tome and Principe	RPJ00IQK5QN	2021-01-25	NULL
6	202	Ezekiel	Marshall Islands	LAY34JKI6HF	2021-01-25	NULL
7	203	Wendy	Heard Island and Mcdonald Islands	FBP43LJW0TM	2021-01-25	NULL

User story3:

As a player, I want to be able to unregister from one or more games.

Assumed this record will with `"Game_Subscription_End_date": "date"` from through UI. This will be consumed as CDC record for that player and necessary update will happen at DB level by using data-pipelines. So associated record status will update `Game_Subscription_End_date` to requested dated in Player_Games table.



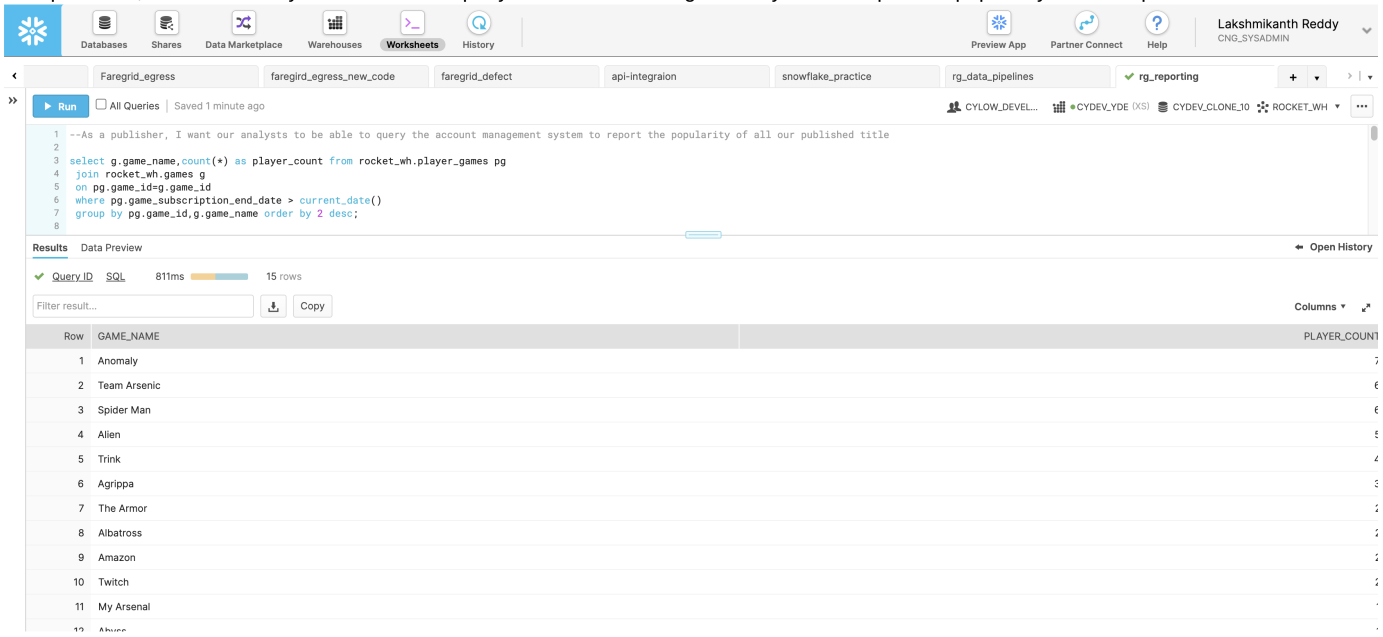
```
select *from rocket_stg.studio_games_stg;


select *from rocket_wh.Player;
select *from rocket_wh.studio;
select *from rocket_wh.games;
select *from rocket_wh.player_games;
```

Query ID SQL 713ms 45 rows

Row	PLAYER_ID	GAME_ID	GAME_SUBSCRIPTION_START_DATE	GAME_SUBSCRIPTION_END_DATE
1	198	1091	2020-11-28	2025-07-11
2	199	1113	2016-10-01	2021-03-23
3	200	1106	2012-04-22	2021-07-19
4	201	1106	2019-01-26	2025-05-14
5	202	1093	2013-08-18	2021-09-05
6	203	1095	2014-02-17	2024-03-26
7	204	1114	2017-03-29	2027-09-04
8	205	1099	2020-02-25	2022-01-11

User story4:

As a publisher, I want our analysts to be able to query the account management system to report the popularity of all our published title.



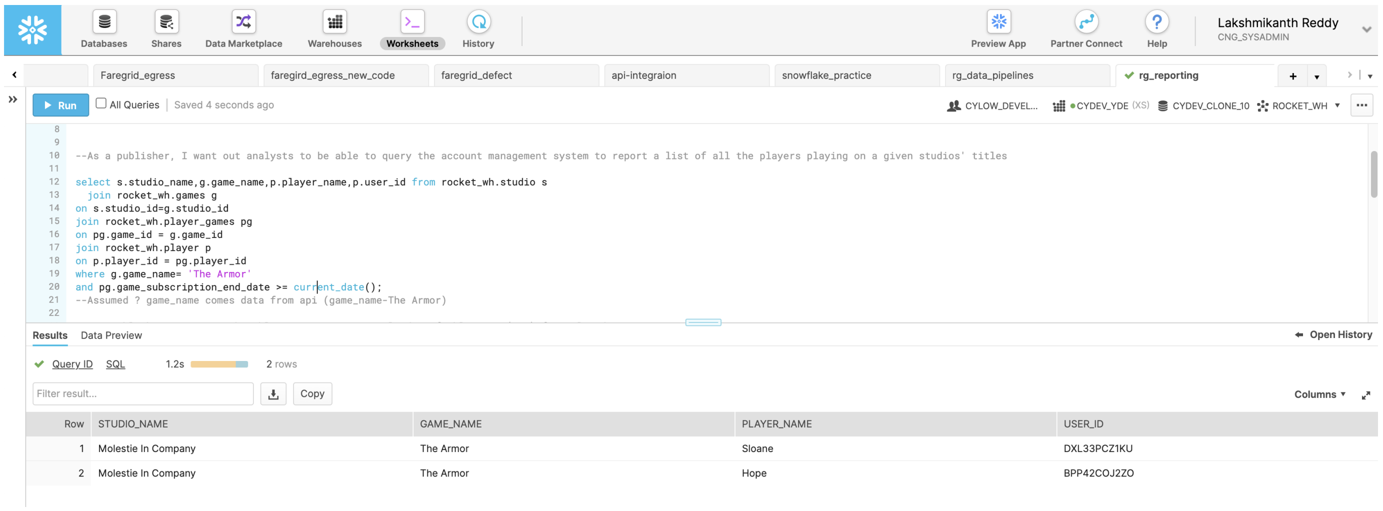
```
--As a publisher, I want our analysts to be able to query the account management system to report the popularity of all our published title

select g.game_name,count(*) as player_count from rocket_wh.player_games pg
 join rocket_wh.games g
 on pg.game_id=g.game_id
 where pg.game_subscription_end_date > current_date()
 group by pg.game_id,g.game_name order by 2 desc;
```

811ms 15 rows

Row	GAME_NAME	PLAYER_COUNT
1	Anomaly	7
2	Team Arsenic	6
3	Spider Man	6
4	Alien	5
5	Trink	4
6	Agrippa	3
7	The Armor	2
8	Albatross	2
9	Amazon	2
10	Twitch	2
11	My Arsenal	1
12	Abyss	

User story5:

As a publisher, I want out analysts to be able to query the account management system to report a list of all the players playing on a given studios' titles



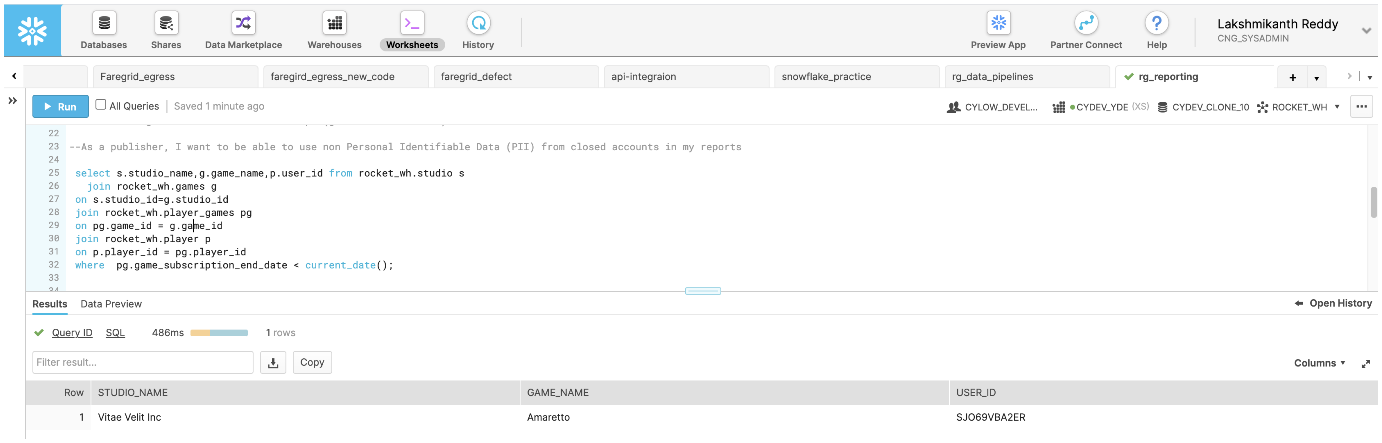
```


--As a publisher, I want out analysts to be able to query the account management system to report a list of all the players playing on a given studios' titles

select s.studio_name,g.game_name,p.player_name,p.user_id from rocket_wh.studio s
  join rocket_wh.games g
on s.studio_id=g.studio_id
join rocket_wh.player_games pg
on pg.game_id = g.game_id
join rocket_wh.player p
on p.player_id = pg.player_id
where g.game_name= 'The Armor'
and pg.game_subscription_end_date >= current_date();
--Assumed ? game_name comes data from api (game_name-The Armor)
```

1.2s 2 rows

Row	STUDIO_NAME	GAME_NAME	PLAYER_NAME	USER_ID
1	Molestie In Company	The Armor	Sloane	DXL33PCZ1KU
2	Molestie In Company	The Armor	Hope	BPP42COJ2ZO

User story6:

As a publisher, I want to be able to use non Personal Identifiable Data (PII) from closed accounts in my reports



```
--As a publisher, I want to be able to use non Personal Identifiable Data (PII) from closed accounts in my reports

 select s.studio_name,g.game_name,p.user_id from rocket_wh.studio s
   join rocket_wh.games g
 on s.studio_id=g.studio_id
 join rocket_wh.player_games pg
 on pg.game_id = g.game_id
 join rocket_wh.player p
 on p.player_id = pg.player_id
 where  pg.game_subscription_end_date < current_date();
```

Row	STUDIO_NAME	GAME_NAME	USER_ID
1	Vitae Velit Inc	Amaretto	SJO69VBA2ER

User story7:

As a publisher, I don't want players seeing any back end reports.

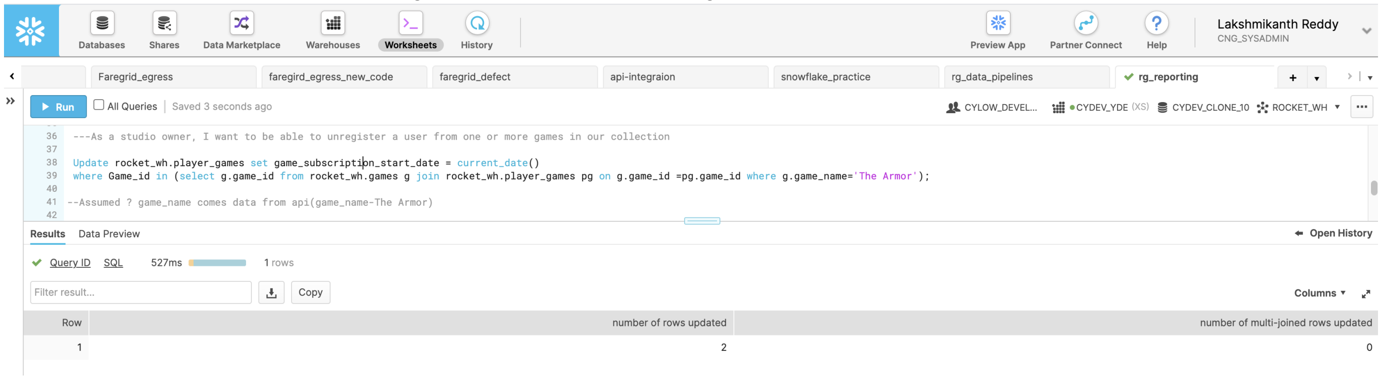
This can be achieved at 3 levels making the following restrictions.

- Do not provide the access to other APIs other than player. Means no access to Studio and publisher APIs.
- Implemented at API gateway level only POST method for ingesting data. Hence no possibility to access other API gateways.
- Finally, we allocated specific DB roles at snowflake level for each user. This will also restrict the access.

FYI.. DB roles are create for `rg_publisher` and `rg_studio`. This can be verified at deploy_roles.sql file provided.

User story8:

As a studio owner, I want to be able to unregister a user from one or more games in our collection

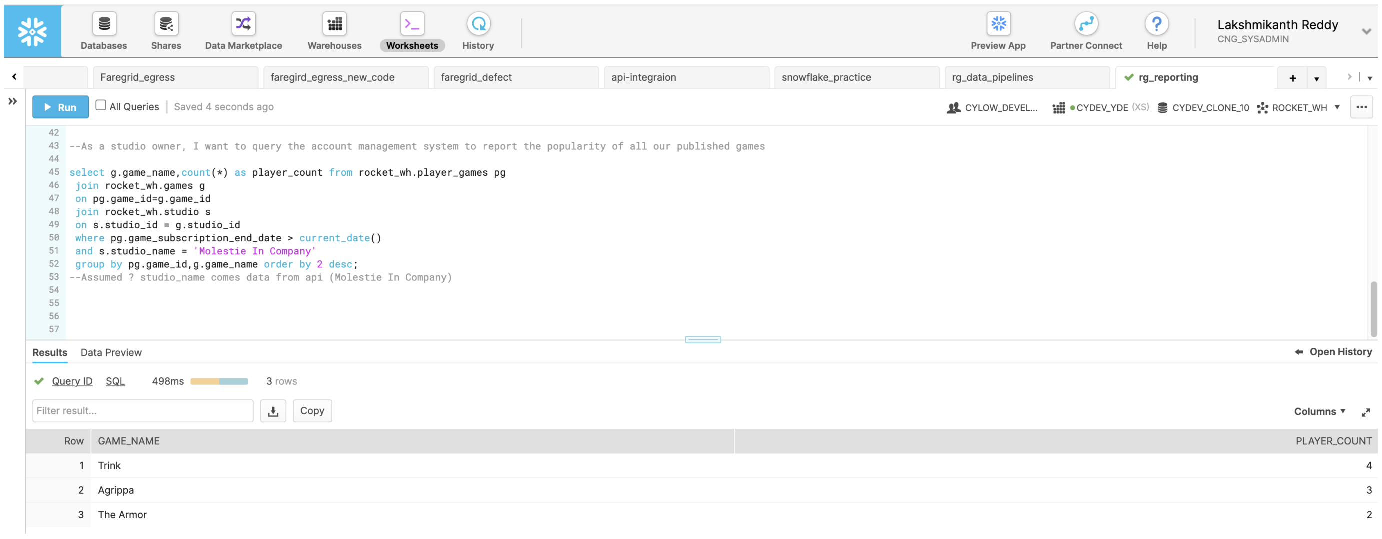


```
---As a studio owner, I want to be able to unregister a user from one or more games in our collection

Update rocket_wh.player_games set game_subscription_start_date = current_date()
where Game_id in (select g.game_id from rocket_wh.games g join rocket_wh.player_games pg on g.game_id =pg.game_id where g.game_name='The Armor');

--Assumed ? game_name comes data from api(game_name-The Armor)
```

Row	number of rows updated	number of multi-joined rows updated
1	2	0

User story9:

As a studio owner, I want to query the account management system to report the popularity of all our published games.



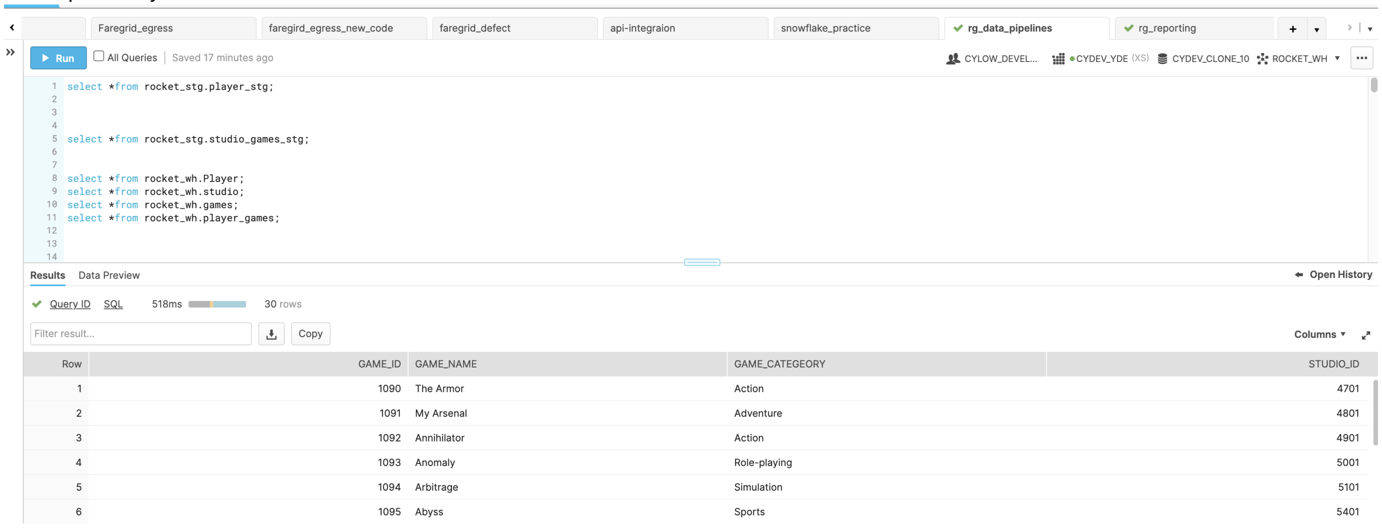
```
--As a studio owner, I want to query the account management system to report the popularity of all our published games

select g.game_name,count(*) as player_count from rocket_wh.player_games pg
 join rocket_wh.games g
 on pg.game_id=g.game_id
 join rocket_wh.studio s
 on s.studio_id = g.studio_id
 where pg.game_subscription_end_date > current_date()
 and s.studio_name = 'Molestie In Company'
 group by pg.game_id,g.game_name order by 2 desc;
--Assumed ? studio_name comes data from api (Molestie In Company)
```

Row	GAME_NAME	PLAYER_COUNT
1	Trink	4
2	Agrippa	3
3	The Armor	2

User story10:

As a studio owner, I don't want other studios to be able to view reports on my performance or my games' popularity.

This can be achieved via API restriction by providing unique studio ID for individual studio-owners. This means only studio owner can access their reports only w.r.t Games.



```
select *from rocket_stg.player_stg;



select *from rocket_stg.studio_games_stg;


select *from rocket_wh.Player;
select *from rocket_wh.studio;
select *from rocket_wh.games;
select *from rocket_wh.player_games;
```

Row	GAME_ID	GAME_NAME	GAME_CATEGEORY	STUDIO_ID
1	1090	The Armor	Action	4701
2	1091	My Arsenal	Adventure	4801
3	1092	Annihilator	Action	4901
4	1093	Anomaly	Role-playing	5001
5	1094	Arbitrage	Simulation	5101
6	1095	Abyss	Sports	5401