

P4

Generate Data takes 11-13min.

P5 Aufgabe 1

ShowPlayer() by name:

start: 11:00:43 end: 11:01:14 average is at: 27 sec

NumberOfGamesPlayed():

Start: 11:11:34 end: 11:11:35 average is at: 0.89 sec

selectedCategories():

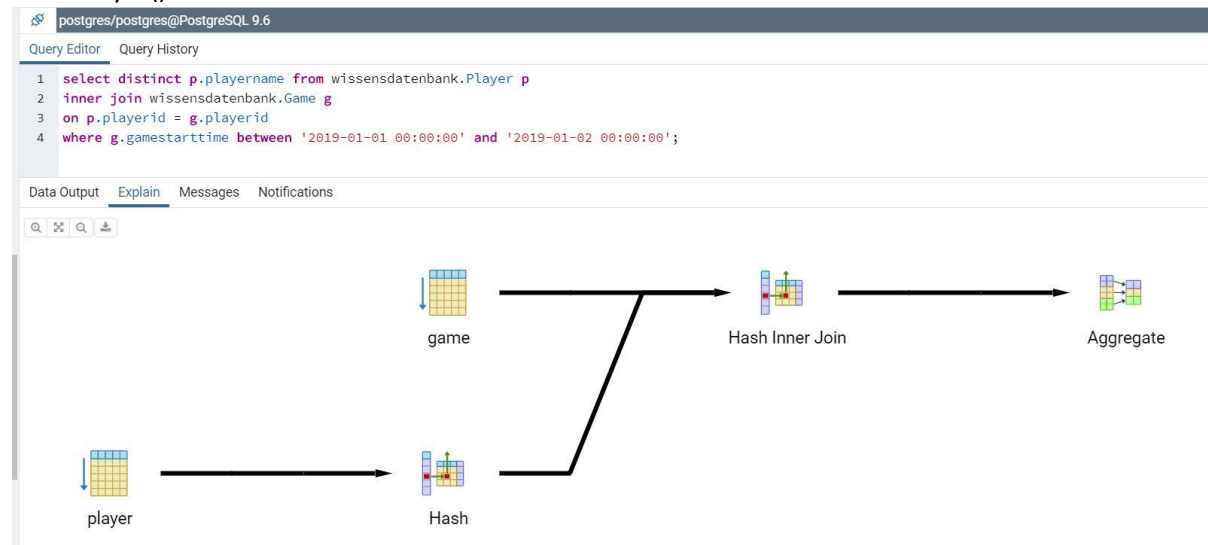
start: 11:12:36 end: 11:12:42 average is at: 5.47sec

showGame():

start: 11:13:58 end: 11:15:04 average is at: 61 sec

P5 Aufgabe 2

showPlayer():



Player joins game which are played during given time period.

showNumberOfGame():

postgres/postgres@PostgreSQL 9.6

Query Editor

```
1 select distinct p.playerid, count(g.gameid)
2 from wissensdatenbank.Game g
3 inner join wissensdatenbank.Player p on p.playerid = g.playerid
4 group by p.playerid order by count(g.gameid);
```

Explain

```
graph LR
    game[game] --> HJ1[Hash Inner Join]
    player[player] --> Hash[Hash]
    Hash --> HJ1
    HJ1 --> Agg[Aggregate]
    Agg --> Sort[Sort]
    Sort --> Unique[Unique]
```

Player join Game, group player for accumulate games. Then sort.

selectedCategories():

Query Editor Query History

```
1 select c.categoryid, count(c.categoryid)
2 from wissensdatenbank.Question q
3 inner join wissensdatenbank.Category c
4 on c.categoryid = q.category_categoryid, wissensdatenbank.Game g,
5 wissensdatenbank.game_question gq
6 where q.question_id in(gq.questions_question_id)
7 and g.gameid = gq.game_gameid
8 group by c.categoryid order by count(c.categoryid);
```

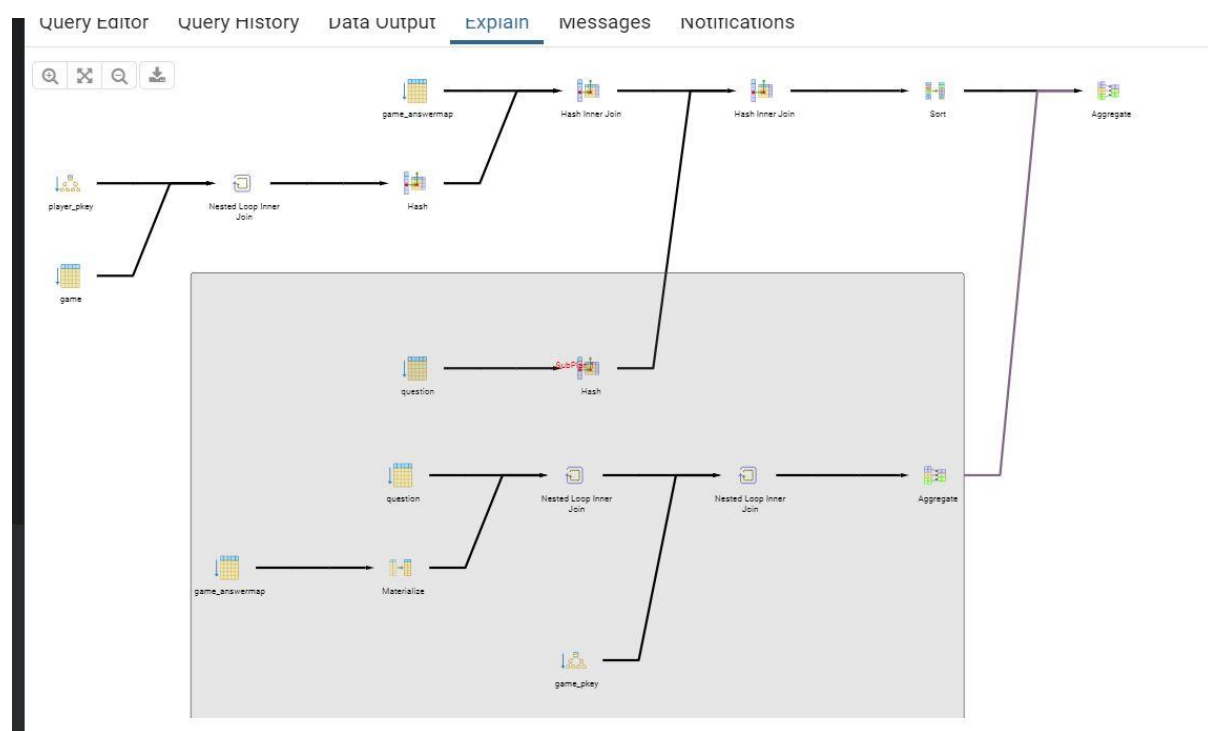
Query Editor Query History Data Output Explain Messages Notifications

```
graph LR
    category[category] --> Hash1[Hash]
    question[question] --> HJ1[Hash Inner Join]
    Hash1 --> HJ1
    HJ1 --> Hash2[Hash]
    game[game] --> Hash3[Hash]
    game_question[game_question] --> HJ2[Hash Inner Join]
    Hash2 --> HJ2
    Hash3 --> HJ2
    HJ2 --> Agg[Aggregate]
    Agg --> Sort[Sort]
```

First join category and question which are played in a game.

showGame():

```
Query Editor  Query History  Data Output  Explain  Messages  Notifications
1  select g.gameid, g.gamestarttime, count(q.question_id),
2  (select count(q1.question_id) from
3  wissensdatenbank.Game g1 inner join wissensdatenbank.game_answermap a1
4  on g1.gameid = a1.game_gameid,
5  wissensdatenbank.Question q1
6  where q1.question_id = a1.question and
7  q1.correct_answer = a1.answer and
8  g.gameid = g1.gameid)
9  from wissensdatenbank.Game g inner join wissensdatenbank.player p
10 on g.playerid = p.playerid
11 inner join wissensdatenbank.game_answermap a
12 on g.gameid = a.game_gameid,
13 wissensdatenbank.Question q where q.question_id = a.question and
14 p.playerid = 3
15 group by g.gameid
```



Player joins game which joins question. Get answer from question. Compare player answer with correct answer. Sort after gameid.

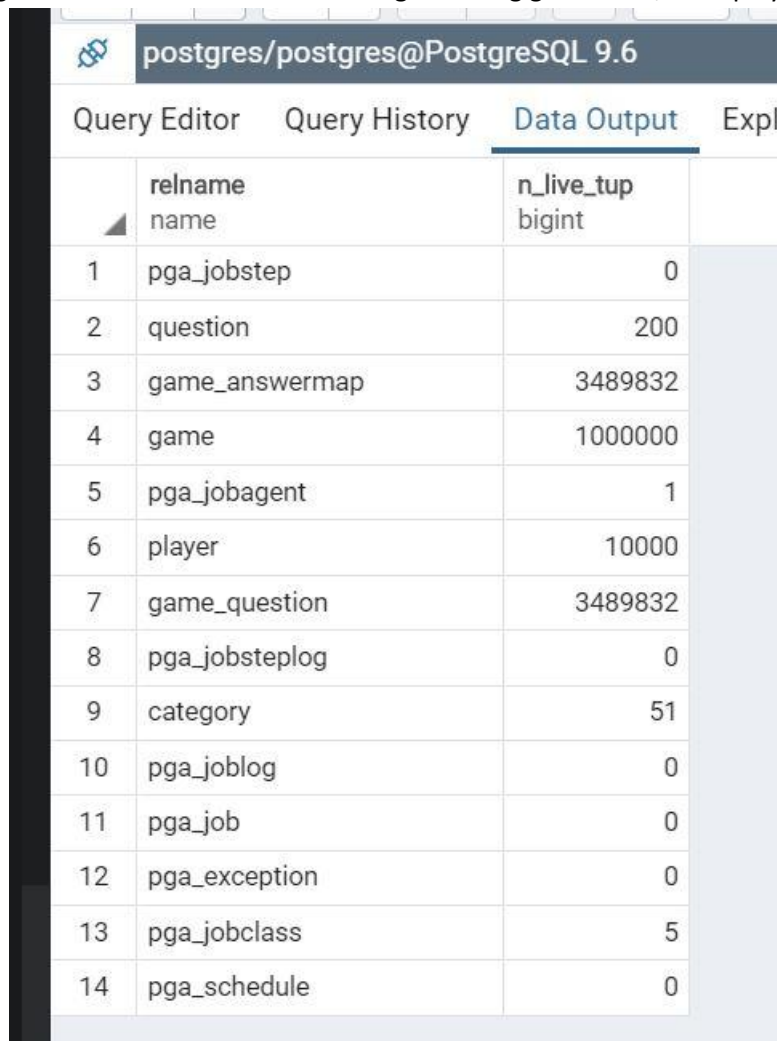
P5 Aufgabe 3

```
select * from pg_indexes where schemaname = 'wissensdatenbank';
```

	Query Editor	Query History	Data Output	Explain	Messages	Notifications
	schemaname name	tablename name	indexname name	tablespace name	indexdef text	
1	wissensdatenba...	category	category_pkey	[null]	CREATE U...	
2	wissensdatenba...	category	category_category_name_key	[null]	CREATE U...	
3	wissensdatenba...	question	question_pkey	[null]	CREATE U...	
4	wissensdatenba...	player	player_pkey	[null]	CREATE U...	
5	wissensdatenba...	player	player_playername_key	[null]	CREATE U...	
6	wissensdatenba...	game	game_pkey	[null]	CREATE U...	
7	wissensdatenba...	game_questi...	game_question_pkey	[null]	CREATE U...	

```
select relname, n_live_tup from pg_stat_user_tables;
```

All stats are expected as it is. We have 10k players, each player plays 100 games. Number of total games should be 1 million. While generating game data, each played game has 2 categories. Each



The screenshot shows a PostgreSQL 9.6 interface with a table view. The table has 14 rows and 2 columns: 'relname' and 'n_live_tup'. The 'relname' column contains names of database tables, and the 'n_live_tup' column contains the number of live tuples (rows) in each table. The table is sorted by 'n_live_tup' in descending order.

	relname name	n_live_tup bigint
1	pga_jobstep	0
2	question	200
3	game_answermap	3489832
4	game	1000000
5	pga_jobagent	1
6	player	10000
7	game_question	3489832
8	pga_jobsteplog	0
9	category	51
10	pga_joblog	0
11	pga_job	0
12	pga_exception	0
13	pga_jobclass	5
14	pga_schedule	0

category has 2 questions. But some categories has only 1 question. That's why the total questions in-game should be between 3 to 4 million entries.

P5 A3:

Didn't need to improve performance through new indexes, because I coded it performance friendly right away.