SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE

Fakulta informatiky a informačných technológii

Databázové systémy

Zadanie 6 - ORM

Tomáš Socha

ID: 110896

Akademický rok 2021/2022

1. Dotazy

a. Patches

- Vlastný dotaz

```
with tabulka as(
    select name as patch_version, extract(EPOCH from release_date)::integer as patch_start_date,
    LEAD(extract(EPOCH from release_date)::integer,1)over (order by name) as patch_end_date
    from patches  p )
select patch_version,patch_start_date, patch_end_date,m.id as match_id,Round(m.duration::numeric/60,2)::real as duration from tabulka
left join matches m
on (m.start_time >= patch_start_date  and (m.start_time < patch_end_date or patch_end_date is NULL))
order by patch_version asc
```

- Vygenerovaný ORM dotaz

```
SELECT anon_1.patch_version, anon_1.patch_start_date, anon_1.patch_end_date, matches.id AS match_id,
CAST(round(CAST(matches.duration AS NUMERIC) / 60, 2) AS REAL) AS duration
FROM (
    SELECT patches.id AS id, patches.name AS patch_version,
    CAST(EXTRACT(epoch FROM patches.release_date) AS INTEGER)
    AS patch_start_date,
    lead(CAST(EXTRACT(epoch FROM patches.release_date) AS INTEGER), 1)
    OVER (PARTITION BY 0 ORDER BY patches.id)
    AS patch_end_date
FROM patches) AS anon_1
LEFT OUTER JOIN matches
ON matches.start_time >= anon_1.patch_start_date
AND (matches.start_time < anon_1.patch_end_date OR anon_1.patch_end_date IS NULL)
ORDER BY anon_1.patch_version
```

b. game_exp

- Vlastný dotaz

```
select p.id, Coalesce(p.nick,'unknown') as player_nick, m.id as match_id, h.localized_name as hero_localized_name,
coalesce(mpd.xp_hero,0)+coalesce(mpd.xp_creep,0)+coalesce(mpd.xp_other,0)+coalesce(mpd.xp_roshan,0) as experiences_gained,
Round(m.duration::numeric/60,2)::real as match_duration_minutes,level as level_gained,
CASE when  player_slot <5 then m.radiant_win else not m.radiant_win end as winner
from players p
left join matches_players_details mpd
on p.id = mpd.player_id
join heroes h
on h.id = mpd.hero_id
left join matches m
on m.id = mpd.match_id
where p.id = <player_id>
order by m.id
```

- Vygenerovaný ORM dotaz

```
SELECT players.id AS id, coalesce(players.nick, 'unknown') AS player_nick, matches.id AS match_id,
heroes.localized_name AS hero_localized_name,
coalesce(matches_players_details.xp_hero, 0) + coalesce(matches_players_details.xp_creep, 0) +
coalesce(matches_players_details.xp_other, 0) + coalesce(matches_players_details.xp_roshan, 0)
AS anon_1,
CAST(round(CAST(matches.duration AS NUMERIC) / 60, 2) AS REAL) AS match_duration_minutes,
matches_players_details.level AS level_gained,
CASE WHEN (matches_players_details.player_slot < 5) THEN matches.radiant_win
ELSE NOT matches.radiant_win END AS winner
FROM players
JOIN matches_players_details
ON players.id = matches_players_details.player_id
JOIN heroes
ON heroes.id = matches_players_details.hero_id
JOIN matches ON matches.id = matches_players_details.match_id
WHERE players.id = '14944' ORDER BY matches.id
```

c. game_objectives

- Vlastný dotaz

```
select p.id, Coalesce(p.nick,'unknown') as player_nick, m.id as match_id, h.localized_name as hero_localized_name,
coalesce(go.subtype,'NO_ACTION')as hero_action, count(*) from players p
left join matches_players_details mpd
on p.id = mpd.player_id
join heroes h
on h.id = mpd.hero_id
left join matches m
on m.id = mpd.match_id
left join game_objectives go
on mpd.id = match_player_detail_id_1
where p.id = <player_id>
group by hero_action,p.id,h.localized_name,m.id
order by match_id
```

- Vygenerovaný ORM dotaz

```
SELECT players.id AS id, coalesce(players.nick, 'unknown') AS player_nick, matches.id AS match_id,
heroes.localized_name AS hero_localized_name,
coalesce(game_objectives.subtype, 'NO_ACTION') AS hero_action, count(*) AS count_1
FROM players
JOIN matches_players_details
ON players.id = matches_players_details.player_id
JOIN heroes
ON heroes.id = matches_players_details.hero_id
JOIN matches
ON matches.id = matches_players_details.match_id
LEFT OUTER JOIN game_objectives
ON game_objectives.match_player_detail_id_1 = matches_players_details.id
WHERE players.id = '14944'
GROUP BY hero_action, players.id, hero_localized_name, matches.id
ORDER BY matches.id
```

d. abilities

- Vlastný dotaz

```
select p.id, Coalesce(p.nick,'unknown')as player_nick,m.id as match_id, h.localized_name as hero_localized_name,
a.name as ability_name, count(a.name), MAX(au.level) as upgrade_level
from players p
left join matches_players_details mpd
on p.id = mpd.player_id
join heroes h
on hero_id = h.id
join matches m
on match_id = m.id
left join ability_upgrades au
on match_player_detail_id = mpd.id
left join abilities a
on ability_id = a.id
where p.id = <player_id>
group by a.name,p.id,p.nick,h.localized_name,m.id
order by m.id
```

- Vygenerovaný ORM dotaz

```
SELECT players.id AS id, coalesce(players.nick, 'unknown') AS player_nick, matches.id AS match_id,
heroes.localized_name AS hero_localized_name,
abilities.name AS ability_name, count(abilities.name) AS count_1, max(ability_upgrades.level) AS max_1
FROM players
JOIN matches_players_details
ON players.id = matches_players_details.player_id
JOIN heroes
ON heroes.id = matches_players_details.hero_id
JOIN matches
ON matches.id = matches_players_details.match_id
JOIN ability_upgrades
ON matches_players_details.id = ability_upgrades.match_player_detail_id
JOIN abilities
ON abilities.id = ability_upgrades.ability_id
WHERE players.id = '14944'
GROUP BY ability_name, players.id, player_nick, hero_localized_name, matches.id
ORDER BY matches.id
```

e. top_purchases

- Vlastný dotaz

```
with tabulka as(
    select item_id,i.name as i_name,hero_id,localized_name as h_name, count(*) as pocet,
    row_number() over(partition by hero_id order by count(*) desc, i.name) from matches m
    left join matches_players_details mpd
    on m.id = match_id
    left join purchase_logs pl
    on mpd.id = match_player_detail_id
    left join items i
    on i.id = pl.item_id
    left join heroes h
    on h.id = hero_id
    where match_id = <match_id>
    and ((player_slot < 5 and radiant_win = true) or (player_slot > 127 and radiant_win = false))
    group by hero_id,localized_name,item_id,i.name order by hero_id, count(*) desc, i.name)
select hero_id as id,h_name as name, pocet as count , item_id as id, i_name as name from tabulka
where row_number <=5
```

- Vygenerovaný ORM dotaz

```
SELECT anon_1.hero_id AS id, anon_1.h_name AS name, anon_1.pocet AS count, anon_1.item_id AS id, anon_1.i_name
AS name
FROM (
    SELECT items.id AS item_id, items.name AS i_name, heroes.id AS hero_id, heroes.localized_name AS h_name, count(*)
    AS pocet,
    row_number() OVER (PARTITION BY heroes.id ORDER BY count(*) DESC, items.name) AS row_number
FROM matches
    JOIN matches_players_details
    ON matches.id = matches_players_details.match_id
    JOIN purchase_logs
    ON matches_players_details.id = purchase_logs.match_player_detail_id
    JOIN items
    ON items.id = purchase_logs.item_id
    JOIN heroes
    ON heroes.id = matches_players_details.hero_id
    WHERE matches.id = '4'
    AND (matches_players_details.player_slot < 5 AND matches.radiant_win = true OR matches_players_details.player_slot > 127
         AND matches.radiant_win = false)
    GROUP BY heroes.id, heroes.localized_name, items.id
    ORDER BY heroes.id, count(*) DESC, items.name) AS anon_1
WHERE anon_1.row_number <= 5
ORDER BY anon_1.hero_id
```

f. ability_usage

- Vlastný dotaz

```
with tabulka as(
    select h.id,h.localized_name as h_name,a.name,
    case when Round(au.time::numeric/m.duration*100::numeric,5) < 10 then '0-9'
    when Round(au.time::numeric/m.duration*100::numeric,5) < 20 then '10-19'
    when Round(au.time::numeric/m.duration*100::numeric,5) < 30 then '20-29'
    when Round(au.time::numeric/m.duration*100::numeric,5) < 40 then '30-39'
    when Round(au.time::numeric/m.duration*100::numeric,5) < 50 then '40-49'
    when Round(au.time::numeric/m.duration*100::numeric,5) < 60 then '50-59'
    when Round(au.time::numeric/m.duration*100::numeric,5) < 70 then '60-69'
    when Round(au.time::numeric/m.duration*100::numeric,5) < 80 then '70-79'
    when Round(au.time::numeric/m.duration*100::numeric,5) < 90 then '80-89'
    when Round(au.time::numeric/m.duration*100::numeric,5) < 100 then '90-99'
    else '100-109' end as bucket,
    CASE when  player_slot < 5 then m.radiant_win else not m.radiant_win end as winner,count (*),
    row_number() over (partition by h.id,CASE when  player_slot < 5 then m.radiant_win
                       else not m.radiant_win end order by count(*) desc)
    from abilities a
    left join ability_upgrades au
    on a.id = ability_id
    left join matches_players_details mpd
    on mpd.id = match_player_detail_id
    left join matches m
    on m.id = match_id
    left join heroes h
    on hero_id = h.id
    where ability_id = <ability_id>
    group by h.id,bucket, winner, a.name
    order by h_name, count desc)
select name,id,h_name as name,winner,bucket,count from tabulka
where row_number = 1
```

- Vygenerovaný ORM dotaz

```
SELECT anon_1.name, anon_1.id, anon_1.h_name AS name, anon_1.winner, anon_1.bucket, anon_1.count
FROM (
    SELECT heroes.id AS id, heroes.localized_name AS h_name, abilities.name AS name,
    CASE WHEN (round((CAST(ability_upgrades.time AS NUMERIC) / CAST(matches.duration AS NUMERIC)) * 100, 5) < 10) THEN '0-9'
    WHEN (round((CAST(ability_upgrades.time AS NUMERIC) / CAST(matches.duration AS NUMERIC)) * 100, 5) < 20) THEN '10-19'
    WHEN (round((CAST(ability_upgrades.time AS NUMERIC) / CAST(matches.duration AS NUMERIC)) * 100, 5) < 30) THEN '20-29'
    WHEN (round((CAST(ability_upgrades.time AS NUMERIC) / CAST(matches.duration AS NUMERIC)) * 100, 5) < 40) THEN '30-39'
    WHEN (round((CAST(ability_upgrades.time AS NUMERIC) / CAST(matches.duration AS NUMERIC)) * 100, 5) < 50) THEN '40-49'
    WHEN (round((CAST(ability_upgrades.time AS NUMERIC) / CAST(matches.duration AS NUMERIC)) * 100, 5) < 60) THEN '50-59'
    WHEN (round((CAST(ability_upgrades.time AS NUMERIC) / CAST(matches.duration AS NUMERIC)) * 100, 5) < 70) THEN '60-69'
    WHEN (round((CAST(ability_upgrades.time AS NUMERIC) / CAST(matches.duration AS NUMERIC)) * 100, 5) < 80) THEN '70-79'
    WHEN (round((CAST(ability_upgrades.time AS NUMERIC) / CAST(matches.duration AS NUMERIC)) * 100, 5) < 90) THEN '80-89'
    WHEN (round((CAST(ability_upgrades.time AS NUMERIC) / CAST(matches.duration AS NUMERIC)) * 100, 5) < 100) THEN '90-99'
    ELSE '100-109' END AS bucket,
    CASE WHEN (matches_players_details.player_slot < 5) THEN matches.radiant_win ELSE NOT matches.radiant_win END AS winner,
    count(*) AS count,
    row_number() OVER (PARTITION BY heroes.id, CASE WHEN (matches_players_details.player_slot < 5) THEN matches.radiant_win
                       ELSE NOT matches.radiant_win END
                       ORDER BY count(*) DESC) AS row_number
    FROM abilities
    JOIN ability_upgrades
    ON abilities.id = ability_upgrades.ability_id
    JOIN matches_players_details
    ON matches_players_details.id = ability_upgrades.match_player_detail_id
    JOIN matches
    ON matches.id = matches_players_details.match_id
    JOIN heroes
    ON heroes.id = matches_players_details.hero_id
    WHERE abilities.id = '5004'
    GROUP BY heroes.id, winner, bucket, abilities.name ñ
    ORDER BY heroes.name, count(*) DESC) AS anon_1
WHERE anon_1.row_number = 1
```

g. tower_kills

- Vlastný dotaz

```
with tabulka2 as(
    with tabulka as(
        select h.id,h.localized_name, match_id, hero_id, lead(hero_id,1) over (order by match_id,go.time )
        as lead_hero_id,
        lead(match_id,1) over (order by match_id,go.time) as lead_match_id,
        row_number() over( order by match_id,hero_id,go.time) - row_number() over( order by match_id,go.time)
        as sequence_id
        from game_objectives go
        left join matches_players_details mpd
        on mpd.id = match_player_detail_id_1
        left join heroes h
        on hero_id = h.id
        where subtype = 'CHAT_MESSAGE_TOWER_KILL' and match_player_detail_id_1 is  not null
        order by match_id,go.time,hero_id)
    select * ,count(*)+1 as tower_kills, row_number() over(partition by localized_name order by count(*)+1 desc)
    from tabulka
    where hero_id = lead_hero_id and match_id = lead_match_id
    group by tabulka.id,localized_name, match_id,hero_id,lead_hero_id, lead_match_id,sequence_id
    order by match_id)
select id,localized_name as name,tower_kills
from tabulka2
where row_number = 1
order by tower_kills desc, name asc
```

- Vygenerovaný ORM dotaz

```
SELECT anon_1.id, anon_1.localized_name AS name, anon_1.tower_kills
FROM (
    SELECT anon_2.id AS id, anon_2.localized_name AS localized_name, anon_2.match_id AS match_id, anon_2.hero_id AS hero_id,
    anon_2.lead_hero_id AS lead_hero_id, anon_2.lead_match_id AS lead_match_id, anon_2.sequence_id AS sequence_id, count(*) + 1
    AS tower_kills,
    row_number() OVER (PARTITION BY anon_2.localized_name ORDER BY count(*) + 1 DESC) AS row_number
    FROM (
        SELECT heroes.id AS id, heroes.localized_name AS localized_name, matches_players_details.match_id AS match_id,
        matches_players_details.hero_id AS hero_id,
        lead(heroes.id) OVER (PARTITION BY 0 ORDER BY matches_players_details.match_id,game_objectives.time) AS lead_hero_id,
        lead(matches_players_details.match_id, 1)
        OVER (PARTITION BY 0 ORDER BY matches_players_details.match_id, heroes.id, game_objectives.time) AS lead_match_id,
        row_number() OVER (PARTITION BY 0 ORDER BY matches_players_details.match_id, heroes.id, game_objectives.time) -
        row_number() OVER (PARTITION BY 0 ORDER BY matches_players_details.match_id, game_objectives.time) AS sequence_id
        FROM game_objectives
        LEFT OUTER JOIN matches_players_details
        ON game_objectives.match_player_detail_id_1 = matches_players_details.id
        LEFT OUTER JOIN heroes
        ON heroes.id = matches_players_details.hero_id
        WHERE game_objectives.subtype = 'CHAT_MESSAGE_TOWER_KILL' AND game_objectives.match_player_detail_id_1 IS NOT NULL
        ORDER BY matches_players_details.match_id, game_objectives.time, heroes.id) AS anon_2
    WHERE anon_2.hero_id = anon_2.lead_hero_id AND anon_2.match_id = anon_2.lead_match_id
    GROUP BY anon_2.id, anon_2.localized_name, anon_2.match_id, anon_2.hero_id, anon_2.lead_hero_id, anon_2.lead_match_id, anon_2.sequence_id
    ORDER BY anon_2.match_id) AS anon_1
WHERE anon_1.row_number = 1
ORDER BY anon_1.tower_kills DESC, anon_1.localized_name ASC
```

2. Explain Analyze

1. Patches

- Vlastný dotaz

```
1  Nested Loop Left Join  (cost=1.59..18834.42 rows=106611 width=48) (actual time=2026.670..19787.631 rows=50005 loops=1)
2  [...] Join Filter: ((m.start_time >= ((date_part('epoch'::text, p.release_date))::integer)) AND ((m.start_time < (lead((date_part('epoch'::text, p.release
3  [...] Rows Removed by Join Filter: 900000
4  [...] -> WindowAgg  (cost=1.59..2.12 rows=19 width=40) (actual time=0.547..1.558 rows=19 loops=1)
5  [...] -> Sort  (cost=1.59..1.64 rows=19 width=40) (actual time=0.459..0.678 rows=19 loops=1)
6  [...] Sort Key: p.name
7  [...] Sort Method: quicksort  Memory: 25kB
8  [...] -> Seq Scan on patches p  (cost=0.00..1.19 rows=19 width=40) (actual time=0.020..0.215 rows=19 loops=1)
9  [...] -> Materialize  (cost=0.00..1266.00 rows=50000 width=12) (actual time=0.012..529.997 rows=50000 loops=19)
10 [...] -> Seq Scan on matches m  (cost=0.00..1016.00 rows=50000 width=12) (actual time=0.019..515.267 rows=50000 loops=1)
11 Planning Time: 0.230 ms
12 Execution Time: 20267.901 ms
```

- Vygenerovaný ORM dotaz

```
1  Sort  (cost=27737.55..28004.07 rows=106611 width=48) (actual time=25147.787..26012.993 rows=50005 loops=1)
2  [...] Sort Key: patches.name
3  [...] Sort Method: quicksort  Memory: 5443kB
4  [...] -> Nested Loop Left Join  (cost=1.59..18834.46 rows=106611 width=48) (actual time=2099.556..24511.112 rows=50005 loops=1)
5  [...] Join Filter: ((matches.start_time >= ((date_part('epoch'::text, patches.release_date))::integer)) AND ((matches.start_time < (lead((date_part('epoch'::te
6  [...] Rows Removed by Join Filter: 900000
7  [...] -> WindowAgg  (cost=1.59..2.16 rows=19 width=48) (actual time=0.806..1.756 rows=19 loops=1)
8  [...] -> Sort  (cost=1.59..1.64 rows=19 width=48) (actual time=0.594..0.834 rows=19 loops=1)
9  [...] Sort Key: patches.id
10 [...] Sort Method: quicksort  Memory: 26kB
11 [...] -> Seq Scan on patches  (cost=0.00..1.19 rows=19 width=48) (actual time=0.019..0.221 rows=19 loops=1)
12 [...] -> Materialize  (cost=0.00..1266.00 rows=50000 width=12) (actual time=0.013..656.954 rows=50000 loops=19)
13 [...] -> Seq Scan on matches  (cost=0.00..1016.00 rows=50000 width=12) (actual time=0.034..534.699 rows=50000 loops=1)
14 Planning Time: 0.551 ms
15 Execution Time: 27124.547 ms
```

Vlastný dotaz obsahuje 12 krokov, zatiaľ čo vygenerovaný pomocou ORM ich má 15. Prvé 3 kroky sú navyše oproti vlastnému dotazu, a od 4. kroku sú potom oba dotazy rovnaké, ako z hľadiska operácie, tak aj podľa hodnôt *cost* a *rows*. Aj *time* sa líši len minimálne.

Avšak najmä kvôli prvým trom krokom je celkový čas vykonávania vytvoreného dotazu pomocou ORM väčší zhruba o 25 percent.

2. game_exp

- Vlastný dotaz

1	Gather Merge (cost=14228.07..14229.13 rows=9 width=63) (actual time=28.676..32.225 rows=13 loops=1)
2	[...] Workers Planned: 3
3	[...] Workers Launched: 3
4	[...] -> Sort (cost=13228.03..13228.04 rows=3 width=63) (actual time=22.960..23.017 rows=3 loops=4)
5	[...] Sort Key: m.id
6	[...] Sort Method: quicksort Memory: 25kB
7	[...] Worker 0: Sort Method: quicksort Memory: 25kB
8	[...] Worker 1: Sort Method: quicksort Memory: 25kB
9	[...] Worker 2: Sort Method: quicksort Memory: 25kB
10	[...] -> Nested Loop Left Join (cost=4.25..13228.00 rows=3 width=63) (actual time=10.034..22.840 rows=3 loops=4)
11	[...] -> Hash Join (cost=3.96..13203.02 rows=3 width=53) (actual time=9.895..22.500 rows=3 loops=4)
12	[...] Hash Cond: (mpd.hero_id = h.id)
13	[...] -> Nested Loop (cost=0.42..13199.47 rows=3 width=47) (actual time=6.267..18.777 rows=3 loops=4)
14	[...] -> Parallel Seq Scan on matches_players_details mpd (cost=0.00..13174.13 rows=3 width=36) (actual time=6.125..18.378 rows=3 loops=4)
15	[...] Filter: (player_id = 14944)
16	[...] Rows Removed by Filter: 124997
17	[...] -> Index Scan using players_pk on players p (cost=0.42..8.44 rows=1 width=15) (actual time=0.033..0.049 rows=1 loops=13)
18	[...] Index Cond: (id = 14944)
19	[...] -> Hash (cost=2.13..2.13 rows=113 width=14) (actual time=3.475..3.488 rows=113 loops=4)
20	[...] Buckets: 1024 Batches: 1 Memory Usage: 14kB
21	[...] -> Seq Scan on heroes h (cost=0.00..2.13 rows=113 width=14) (actual time=0.087..1.822 rows=113 loops=4)
22	[...] -> Index Scan using matches_pk on matches m (cost=0.29..8.31 rows=1 width=9) (actual time=0.034..0.038 rows=1 loops=13)
23	[...] Index Cond: (id = mpd.match_id)
24	Planning Time: 0.555 ms
25	Execution Time: 32.932 ms

- Vygenerovaný ORM dotaz

1	Gather Merge (cost=14228.07..14229.13 rows=9 width=63) (actual time=27.966..31.094 rows=13 loops=1)
2	[...] Workers Planned: 3
3	[...] Workers Launched: 3
4	[...] -> Sort (cost=13228.03..13228.04 rows=3 width=63) (actual time=22.099..22.153 rows=3 loops=4)
5	[...] Sort Key: matches.id
6	[...] Sort Method: quicksort Memory: 25kB
7	[...] Worker 0: Sort Method: quicksort Memory: 25kB
8	[...] Worker 1: Sort Method: quicksort Memory: 25kB
9	[...] Worker 2: Sort Method: quicksort Memory: 25kB
10	[...] -> Nested Loop (cost=4.25..13228.00 rows=3 width=63) (actual time=8.241..21.992 rows=3 loops=4)
11	[...] -> Hash Join (cost=3.96..13203.02 rows=3 width=53) (actual time=8.115..21.666 rows=3 loops=4)
12	[...] Hash Cond: (matches_players_details.hero_id = heroes.id)
13	[...] -> Nested Loop (cost=0.42..13199.47 rows=3 width=47) (actual time=4.813..18.286 rows=3 loops=4)
14	[...] -> Parallel Seq Scan on matches_players_details (cost=0.00..13174.13 rows=3 width=36) (actual time=4.741..17.981 rows=3 loops=4)
15	[...] Filter: (player_id = 14944)
16	[...] Rows Removed by Filter: 124997
17	[...] -> Index Scan using players_pk on players (cost=0.42..8.44 rows=1 width=15) (actual time=0.023..0.039 rows=1 loops=13)
18	[...] Index Cond: (id = 14944)
19	[...] -> Hash (cost=2.13..2.13 rows=113 width=14) (actual time=3.105..3.116 rows=113 loops=4)
20	[...] Buckets: 1024 Batches: 1 Memory Usage: 14kB
21	[...] -> Seq Scan on heroes (cost=0.00..2.13 rows=113 width=14) (actual time=0.045..1.530 rows=113 loops=4)
22	[...] -> Index Scan using matches_pk on matches (cost=0.29..8.31 rows=1 width=9) (actual time=0.038..0.042 rows=1 loops=13)
23	[...] Index Cond: (id = matches_players_details.match_id)
24	Planning Time: 0.573 ms
25	Execution Time: 31.579 ms

V prípade tohto zadania majú oba dotazy rovnaké operácie, a v rámci nich aj zhodné hodnoty cost, rows a zanedbateľné rozdiely v time. Výsledný čas vyšiel cca o 1ms lepší v prípade dotazu vygenerovaného pomocou ORM.

3. game_objectives

- Vlastný dotaz

1	GroupAggregate (cost=34801.70..34802.28 rows=23 width=90) (actual time=6780.092..6780.613 rows=16 loops=1)
2	[...] Group Key: m.id, (COALESCE(go.subtype, 'NO_ACTION'::text)), p.id, h.localized_name
3	[...] -> Sort (cost=34801.70..34801.76 rows=23 width=61) (actual time=6780.039..6780.219 rows=18 loops=1)
4	[...] Sort Key: m.id, (COALESCE(go.subtype, 'NO_ACTION'::text)), h.localized_name
5	[...] Sort Method: quicksort Memory: 26kB
6	[...] -> Nested Loop (cost=21591.84..34801.18 rows=23 width=61) (actual time=6767.114..6779.835 rows=18 loops=1)
7	[...] -> Index Scan using players_pk on players p (cost=0.42..8.44 rows=1 width=15) (actual time=0.020..0.042 rows=1 loops=1)
8	[...] Index Cond: (id = 14944)
9	[...] -> Gather (cost=21591.42..34792.51 rows=23 width=42) (actual time=6767.054..6787.665 rows=18 loops=1)
10	[...] Workers Planned: 3
11	[...] Workers Launched: 3
12	[...] -> Parallel Hash Left Join (cost=20591.42..33790.21 rows=7 width=42) (actual time=6755.577..6766.663 rows=4 loops=4)
13	[...] Hash Cond: (mpd.id = go.match_player_detail_id_1)
14	[...] -> Nested Loop Left Join (cost=3.83..13202.60 rows=3 width=22) (actual time=8.633..19.203 rows=3 loops=4)
15	[...] -> Hash Join (cost=3.54..13177.68 rows=3 width=22) (actual time=8.528..18.928 rows=3 loops=4)
16	[...] Hash Cond: (mpd.hero_id = h.id)
17	[...] -> Parallel Seq Scan on matches_players_details mpd (cost=0.00..13174.13 rows=3 width=16) (actual time=6.082..16.402 rows=3 loops=4)
18	[...] Filter: (player_id = 14944)
19	[...] Rows Removed by Filter: 124997
20	[...] -> Hash (cost=2.13..2.13 rows=113 width=14) (actual time=2.337..2.347 rows=113 loops=4)
21	[...] Buckets: 1024 Batches: 1 Memory Usage: 14kB
22	[...] -> Seq Scan on heroes h (cost=0.00..2.13 rows=113 width=14) (actual time=0.063..1.179 rows=113 loops=4)
23	[...] -> Index Only Scan using matches_pk on matches m (cost=0.29..8.31 rows=1 width=4) (actual time=0.042..0.045 rows=1 loops=13)
24	[...] Index Cond: (id = mpd.match_id)
25	[...] Heap Fetches: 13
26	[...] -> Parallel Hash (cost=15856.15..15856.15 rows=378515 width=28) (actual time=6744.325..6744.334 rows=293349 loops=4)
27	[...] Buckets: 2097152 Batches: 1 Memory Usage: 60736kB
28	[...] -> Parallel Seq Scan on game_objectives go (cost=0.00..15856.15 rows=378515 width=28) (actual time=0.032..3367.041 rows=293349 loops=4)
29	Planning Time: 0.742 ms
30	Execution Time: 6789.426 ms

- Vygenerovaný ORM dotaz

```
1  GroupAggregate  (cost=34801.70..34802.28 rows=23 width=90) (actual time=7307.359..7307.873 rows=16 loops=1)
2  [...] Group Key: matches.id, (COALESCE(game_objectives.subtype, 'NO_ACTION'::text)), players.id, heroes.localized_name
3  [...] ->  Sort  (cost=34801.70..34801.76 rows=23 width=61) (actual time=7307.286..7307.471 rows=18 loops=1)
4  [...] Sort Key: matches.id, (COALESCE(game_objectives.subtype, 'NO_ACTION'::text)), heroes.localized_name
5  [...] Sort Method: quicksort  Memory: 26kB
6  [...] ->  Nested Loop  (cost=21591.84..34801.18 rows=23 width=61) (actual time=7288.859..7307.047 rows=18 loops=1)
7  [...] ->  Index Scan using players_pk on players  (cost=0.42..8.44 rows=1 width=15) (actual time=0.020..0.053 rows=1 loops=1)
8  [...] Index Cond: (id = 14944)
9  [...] ->  Gather  (cost=21591.42..34792.51 rows=23 width=42) (actual time=7288.807..7308.426 rows=18 loops=1)
10 [...] Workers Planned: 3
11 [...] Workers Launched: 3
12 [...] ->  Parallel Hash Left Join  (cost=20591.42..33790.21 rows=7 width=42) (actual time=7286.094..7296.809 rows=4 loops=4)
13 [...] Hash Cond: (matches_players_details.id = game_objectives.match_player_detail_id_1)
14 [...] ->  Nested Loop  (cost=3.83..13202.60 rows=3 width=22) (actual time=9.795..20.073 rows=3 loops=4)
15 [...] ->  Hash Join  (cost=3.54..13177.68 rows=3 width=22) (actual time=9.688..19.804 rows=3 loops=4)
16 [...] Hash Cond: (matches_players_details.hero_id = heroes.id)
17 [...] ->  Parallel Seq Scan on matches_players_details  (cost=0.00..13174.13 rows=3 width=16) (actual time=6.572..16.599 rows=3 loops=4)
18 [...] Filter: (player_id = 14944)
19 [...] Rows Removed by Filter: 124997
20 [...] ->  Hash  (cost=2.13..2.13 rows=113 width=14) (actual time=2.444..2.453 rows=113 loops=4)
21 [...] Buckets: 1024  Batches: 1  Memory Usage: 14kB
22 [...] ->  Seq Scan on heroes  (cost=0.00..2.13 rows=113 width=14) (actual time=0.056..1.235 rows=113 loops=4)
23 [...] ->  Index Only Scan using matches_pk on matches  (cost=0.29..8.31 rows=1 width=4) (actual time=0.040..0.043 rows=1 loops=13)
24 [...] Index Cond: (id = matches_players_details.match_id)
25 [...] Heap Fetches: 13
26 [...] ->  Parallel Hash  (cost=15856.15..15856.15 rows=378515 width=28) (actual time=7273.709..7273.719 rows=293349 loops=4)
27 [...] Buckets: 2097152  Batches: 1  Memory Usage: 60736kB
28 [...] ->  Parallel Seq Scan on game_objectives  (cost=0.00..15856.15 rows=378515 width=28) (actual time=0.037..3525.202 rows=293349 loops=4)
29 Planning Time: 0.977 ms
30 Execution Time: 7310.287 ms
```

Oba dotazy majú zhodné operácie ako aj atribúty cost a rows. Celkový čas potrebný pre vykonanie dotazu je v tomto prípade menší u môjho dotazu o cca 500ms.

4. abilities

- Vlastný dotaz

1	GroupAggregate (cost=143308.81..143313.73 rows=179 width=95) (actual time=62247.245..62252.808 rows=63 loops=1)
2	[...] Group Key: m.id, a.name, p.id, h.localized_name
3	[...] -> Sort (cost=143308.81..143309.26 rows=179 width=55) (actual time=62247.168..62249.644 rows=239 loops=1)
4	[...] Sort Key: m.id, a.name, h.localized_name
5	[...] Sort Method: quicksort Memory: 49kB
6	[...] -> Nested Loop (cost=119143.77..143302.11 rows=179 width=55) (actual time=61244.037..62244.247 rows=239 loops=1)
7	[...] -> Index Scan using players_pk on players p (cost=0.42..8.44 rows=1 width=15) (actual time=0.021..0.045 rows=1 loops=1)
8	[...] Index Cond: (id = 14944)
9	[...] -> Gather (cost=119143.35..143291.89 rows=179 width=44) (actual time=61243.979..62423.939 rows=239 loops=1)
10	[...] Workers Planned: 3
11	[...] Workers Launched: 3
12	[...] -> Nested Loop Left Join (cost=118143.35..142273.99 rows=58 width=44) (actual time=61767.186..62229.449 rows=60 loops=4)
13	[...] -> Parallel Hash Left Join (cost=118143.08..142257.02 rows=58 width=26) (actual time=61767.115..62226.401 rows=60 loops=4)
14	[...] Hash Cond: (mpd.id = au.match_player_detail_id)
15	[...] -> Nested Loop (cost=3.83..13202.60 rows=3 width=22) (actual time=14.425..25.677 rows=3 loops=4)
16	[...] -> Hash Join (cost=3.54..13177.68 rows=3 width=22) (actual time=14.269..25.315 rows=3 loops=4)
17	[...] Hash Cond: (mpd.hero_id = h.id)
18	[...] -> Parallel Seq Scan on matches_players_details mpd (cost=0.00..13174.13 rows=3 width=16) (actual time=10.459..21.404 rows=3 loops=4)
19	[...] Filter: (player_id = 14944)
20	[...] Rows Removed by Filter: 124997
21	[...] -> Hash (cost=2.13..2.13 rows=113 width=14) (actual time=3.710..3.719 rows=113 loops=4)
22	[...] Buckets: 1024 Batches: 1 Memory Usage: 14kB
23	[...] -> Seq Scan on heroes h (cost=0.00..2.13 rows=113 width=14) (actual time=0.059..1.850 rows=113 loops=4)
24	[...] -> Index Only Scan using matches_pk on matches m (cost=0.29..8.31 rows=1 width=4) (actual time=0.049..0.054 rows=1 loops=13)
25	[...] Index Cond: (id = mpd.match_id)
26	[...] Heap Fetches: 13
27	[...] -> Parallel Hash (cost=79290.00..79290.00 rows=2234900 width=12) (actual time=61141.137..61141.147 rows=2234900 loops=4)
28	[...] Buckets: 4194304 Batches: 8 Memory Usage: 85504kB
29	[...] -> Parallel Seq Scan on ability_upgrades au (cost=0.00..79290.00 rows=2234900 width=12) (actual time=0.052..28886.967 rows=2234900 loops=4)
30	[...] -> Index Scan using abilities_pk on abilities a (cost=0.28..0.29 rows=1 width=26) (actual time=0.016..0.016 rows=1 loops=239)
31	[...] Index Cond: (id = au.ability_id)
32	Planning Time: 1.196 ms
33	Execution Time: 62439.022 ms

- Vygenerovaný ORM dotaz

```
1  GroupAggregate  (cost=99756.31..99761.68 rows=179 width=84) (actual time=43276.299..43281.495 rows=63 loops=1)
2  [...] Group Key: matches.id, abilities.name, players.id, (COALESCE(players.nick, 'unknown'::text)), heroes.localized_name
3  [...] ->  Sort  (cost=99756.31..99756.76 rows=179 width=76) (actual time=43276.215..43278.421 rows=239 loops=1)
4  [...] Sort Key: matches.id, abilities.name, (COALESCE(players.nick, 'unknown'::text)), heroes.localized_name
5  [...] Sort Method: quicksort  Memory: 49kB
6  [...] ->  Nested Loop  (cost=14178.69..99749.61 rows=179 width=76) (actual time=2568.994..43273.441 rows=239 loops=1)
7  [...] ->  Index Scan using players_pk on players  (cost=0.42..8.44 rows=1 width=15) (actual time=0.026..0.063 rows=1 loops=1)
8  [...] Index Cond: (id = 14944)
9  [...] ->  Gather  (cost=14178.27..99739.38 rows=179 width=44) (actual time=2568.926..43268.700 rows=239 loops=1)
10 [...] Workers Planned: 4
11 [...] Workers Launched: 4
12 [...] ->  Nested Loop  (cost=13178.27..98721.48 rows=45 width=44) (actual time=19250.912..43262.521 rows=48 loops=5)
13 [...] ->  Nested Loop  (cost=13178.00..98708.32 rows=45 width=26) (actual time=19250.817..43260.073 rows=48 loops=5)
14 [...] ->  Hash Join  (cost=13177.71..98334.48 rows=45 width=26) (actual time=19250.612..43257.388 rows=48 loops=5)
15 [...] Hash Cond: (matches_players_details.hero_id = heroes.id)
16 [...] ->  Parallel Hash Join  (cost=13174.17..98330.82 rows=45 width=20) (actual time=19247.271..43252.960 rows=48 loops=5)
17 [...] Hash Cond: (ability_upgrades.match_player_detail_id = matches_players_details.id)
18 [...] ->  Parallel Seq Scan on ability_upgrades  (cost=0.00..79290.00 rows=2234900 width=12) (actual time=0.031..21406.938 rows=1787920 loops=5)
19 [...] ->  Parallel Hash  (cost=13174.13..13174.13 rows=3 width=16) (actual time=17.102..17.113 rows=3 loops=5)
20 [...] Buckets: 1024  Batches: 1  Memory Usage: 104kB
21 [...] ->  Parallel Seq Scan on matches_players_details  (cost=0.00..13174.13 rows=3 width=16) (actual time=9.111..16.882 rows=3 loops=5)
22 [...] Filter: (player_id = 14944)
23 [...] Rows Removed by Filter: 99997
24 [...] ->  Hash  (cost=2.13..2.13 rows=113 width=14) (actual time=3.178..3.189 rows=113 loops=5)
25 [...] Buckets: 1024  Batches: 1  Memory Usage: 14kB
26 [...] ->  Seq Scan on heroes  (cost=0.00..2.13 rows=113 width=14) (actual time=0.050..1.615 rows=113 loops=5)
27 [...] ->  Index Only Scan using matches_pk on matches  (cost=0.29..8.31 rows=1 width=4) (actual time=0.020..0.020 rows=1 loops=239)
28 [...] Index Cond: (id = matches_players_details.match_id)
29 [...] Heap Fetches: 239
30 [...] ->  Index Scan using abilities_pk on abilities  (cost=0.28..0.29 rows=1 width=26) (actual time=0.016..0.016 rows=1 loops=239)
31 [...] Index Cond: (id = ability_upgrades.ability_id)
32 Planning Time: 2.640 ms
33 Execution Time: 43282.725 ms
```

Nasledujúce dotazy sa líšia ako v typoch operácií tak aj ich atribútov. Už hneď v prvom kroku je možné si všimnúť rozdiely v atribútoch, kde cost je väčšia pri vlastnej query, počet riadkov je rovnaký, no líši sa počet stĺpcov, čo sa prejavilo aj na výslednom čase. Vo vlastnom dotaze je šírka až 95 kde pri ORM je iba 84. V šiestom kroku obsahuje vlastný dotaz väčšiu hodnotu cost, aj napriek menšiemu width. V deviatom kroku si zas môžeme všimnúť vyššiu hodnotu cost u vlastného dotazu napriek tomu že zvyšné atribúty sú rovnaké. V 12.kroku už pozorujeme taktiež výrazne zvýšenie počtu riadkov u vlastného dotazu.

V nasledujúcich krokoch sa už v jednotlivých krokov vyskytovali aj iné typy operácií. 20., respektíve 23. riadok nám poskytuje informáciu koľko riadkov bolo odstránených filtrom. V dotaze vygenerovanom ORM je to zhruba o 25% menej čo taktiež svedčí o fakte, že orm pracuje v tomto prípade s menšou množinou vďaka čomu sú následne operácie vykonávané rýchlejšie.

Celkový čas potrebný pre ORM dotaz je vo výsledku zhruba o tretinu menší.

5. top_purchases

- Vlastný dotaz

1	Subquery Scan on tabulka (cost=251271.51..251277.88 rows=61 width=40) (actual time=106374.372..106379.520 rows=25 loops=1)
2	[...] Filter: (tabulka.row_number <= 5)
3	[...] Rows Removed by Filter: 103
4	[...] -> WindowAgg (cost=251271.51..251275.61 rows=182 width=48) (actual time=106374.347..106378.081 rows=128 loops=1)
5	[...] -> Sort (cost=251271.51..251271.97 rows=182 width=40) (actual time=106374.280..106375.459 rows=128 loops=1)
6	[...] Sort Key: mpd.hero_id, (count(*)) DESC, i.name
7	[...] Sort Method: quicksort Memory: 35kB
8	[...] -> GroupAggregate (cost=251260.13..251264.68 rows=182 width=40) (actual time=106367.742..106372.861 rows=128 loops=1)
9	[...] Group Key: mpd.hero_id, h.localized_name, pl.item_id, i.name
10	[...] -> Sort (cost=251260.13..251260.59 rows=182 width=32) (actual time=106367.689..106369.542 rows=200 loops=1)
11	[...] Sort Key: mpd.hero_id, h.localized_name, pl.item_id, i.name
12	[...] Sort Method: quicksort Memory: 40kB
13	[...] -> Nested Loop (cost=219464.67..251253.30 rows=182 width=32) (actual time=106298.596..106365.570 rows=200 loops=1)
14	[...] Join Filter: (((mpd.player_slot < 5) AND m.radiant_win) OR ((mpd.player_slot > 127) AND (NOT m.radiant_win)))
15	[...] Rows Removed by Join Filter: 145
16	[...] -> Index Scan using matches_pk on matches m (cost=0.29..8.31 rows=1 width=5) (actual time=0.027..0.051 rows=1 loops=1)
17	[...] Index Cond: (id = 4)
18	[...] Filter: (radiant_win OR (NOT radiant_win))
19	[...] -> Gather (cost=219464.38..251241.17 rows=255 width=40) (actual time=106298.536..106545.910 rows=345 loops=1)
20	[...] Workers Planned: 3
21	[...] Workers Launched: 3
22	[...] -> Hash Left Join (cost=218464.38..250215.67 rows=82 width=40) (actual time=106043.186..106351.704 rows=86 loops=4)
23	[...] Hash Cond: (mpd.hero_id = h.id)
24	[...] -> Hash Left Join (cost=218460.84..250211.90 rows=82 width=30) (actual time=106040.832..106347.656 rows=86 loops=4)
25	[...] Hash Cond: (pl.item_id = i.id)
26	[...] -> Parallel Hash Left Join (cost=218452.81..250203.65 rows=82 width=16) (actual time=106035.302..106340.565 rows=86 loops=4)
27	[...] Hash Cond: (mpd.id = pl.match_player_detail_id)
28	[...] -> Parallel Seq Scan on matches_players_details mpd (cost=0.00..13980.58 rows=2 width=16) (actual time=15.530..22.469 rows=2 loops=4)
29	[...] Filter: ((match_id = 4) AND ((player_slot < 5) OR (player_slot > 127)))
30	[...] Rows Removed by Filter: 124998
31	[...] -> Parallel Hash (cost=143829.36..143829.36 rows=4548436 width=8) (actual time=105691.470..105691.479 rows=4548436 loops=4)
32	[...] Buckets: 4194304 Batches: 8 Memory Usage: 122272kB
33	[...] -> Parallel Seq Scan on purchase_logs pl (cost=0.00..143829.36 rows=4548436 width=8) (actual time=0.029..51721.465 rows=4548436 loops=4)
34	[...] -> Hash (cost=4.68..4.68 rows=268 width=18) (actual time=5.464..5.474 rows=268 loops=4)
35	[...] Buckets: 1024 Batches: 1 Memory Usage: 22kB
36	[...] -> Seq Scan on items i (cost=0.00..4.68 rows=268 width=18) (actual time=0.055..2.718 rows=268 loops=4)
37	[...] -> Hash (cost=2.13..2.13 rows=113 width=14) (actual time=2.294..2.303 rows=113 loops=4)
38	[...] Buckets: 1024 Batches: 1 Memory Usage: 14kB
39	[...] -> Seq Scan on heroes h (cost=0.00..2.13 rows=113 width=14) (actual time=0.043..1.165 rows=113 loops=4)
40	Planning Time: 0.957 ms
41	Execution Time: 106566.378 ms

- Vygenerovaný ORM dotaz

```
1  Subquery Scan on anon_1  (cost=156860.45..156866.82 rows=61 width=40) (actual time=83603.130..83609.484 rows=25 loops=1)
2  [...] Filter: (anon_1.row_number <= 5)
3  [...] Rows Removed by Filter: 103
4  [...] -> WindowAgg  (cost=156860.45..156864.54 rows=182 width=48) (actual time=83603.100..83607.610 rows=128 loops=1)
5  [...] -> Sort  (cost=156860.45..156860.90 rows=182 width=40) (actual time=83603.012..83604.421 rows=128 loops=1)
6  [...] Sort Key: heroes.id, (count(*)) DESC, items.name
7  [...] Sort Method: quicksort  Memory: 35kB
8  [...] -> GroupAggregate  (cost=156849.98..156853.62 rows=182 width=40) (actual time=83592.696..83600.874 rows=128 loops=1)
9  [...] Group Key: heroes.id, items.id
10 [...] -> Sort  (cost=156849.98..156850.43 rows=182 width=32) (actual time=83592.609..83595.500 rows=200 loops=1)
11 [...] Sort Key: heroes.id, items.id
12 [...] Sort Method: quicksort  Memory: 40kB
13 [...] -> Nested Loop  (cost=1036.35..156843.14 rows=182 width=32) (actual time=28.194..83589.865 rows=200 loops=1)
14 [...] Join Filter: (((matches_players_details.player_slot < 5) AND matches.radiant_win) OR ((matches_players_details.player_slot > 127) AND (NOT matches.radiant_win)))
15 [...] Rows Removed by Join Filter: 145
16 [...] -> Index Scan using matches_pk on matches  (cost=0.29..8.31 rows=1 width=5) (actual time=0.019..0.062 rows=1 loops=1)
17 [...] Index Cond: (id = 4)
18 [...] Filter: (radiant_win OR (NOT radiant_win))
19 [...] -> Gather  (cost=1036.06..156831.01 rows=255 width=40) (actual time=28.138..83582.775 rows=345 loops=1)
20 [...] Workers Planned: 4
21 [...] Workers Launched: 4
22 [...] -> Hash Join  (cost=36.06..155805.51 rows=64 width=40) (actual time=33445.146..83579.980 rows=69 loops=5)
23 [...] Hash Cond: (matches_players_details.hero_id = heroes.id)
24 [...] -> Hash Join  (cost=32.52..155801.79 rows=64 width=30) (actual time=33441.689..83574.558 rows=69 loops=5)
25 [...] Hash Cond: (purchase_logs.item_id = items.id)
26 [...] -> Hash Join  (cost=24.49..155793.59 rows=64 width=16) (actual time=33434.418..83565.321 rows=69 loops=5)
27 [...] Hash Cond: (purchase_logs.match_player_detail_id = matches_players_details.id)
28 [...] -> Parallel Seq Scan on purchase_logs  (cost=0.00..143829.36 rows=4548436 width=8) (actual time=0.029..41600.813 rows=3638749 loops=5)
29 [...] -> Hash  (cost=24.40..24.40 rows=7 width=16) (actual time=0.379..0.389 rows=10 loops=5)
30 [...] Buckets: 1024  Batches: 1  Memory Usage: 9kB
31 [...] -> Index Scan using idx_match_id_player_id on matches_players_details  (cost=0.42..24.40 rows=7 width=16) (actual time=0.035..0.217 rows=10 loops=5)
32 [...] Index Cond: (match_id = 4)
33 [...] Filter: ((player_slot < 5) OR (player_slot > 127))
34 [...] -> Hash  (cost=4.68..4.68 rows=268 width=18) (actual time=7.189..7.198 rows=268 loops=5)
35 [...] Buckets: 1024  Batches: 1  Memory Usage: 22kB
36 [...] -> Seq Scan on items  (cost=0.00..4.68 rows=268 width=18) (actual time=0.037..3.518 rows=268 loops=5)
37 [...] -> Hash  (cost=2.13..2.13 rows=113 width=14) (actual time=3.285..3.295 rows=113 loops=5)
38 [...] Buckets: 1024  Batches: 1  Memory Usage: 14kB
39 [...] -> Seq Scan on heroes  (cost=0.00..2.13 rows=113 width=14) (actual time=0.040..1.631 rows=113 loops=5)
40 Planning Time: 1.207 ms
41 Execution Time: 83610.474 ms
```

Porovnávané dotazy sú zhodné v typoch operácií až do 29. kroku kde sa líšia len v cost, ktorý je zakaždým vyšší pri mojom dotaze. Následne si môžeme pri ORM dotaze všimnúť použitia indexov pri filtrovaní položiek podľa match_id, zatiaľ čo v mojom dotaze je použitý klasický filter.

Výsledný čas je v prospech ORM dotazu, ktorý je rýchlejši zhruba o 20%.

6. ability_usage

- Vlastný dotaz

1	Subquery Scan on tabulka (cost=131559.98..132132.11 rows=191 width=77) (actual time=5150.830..5151.288 rows=3 loops=1)
2	[...] Filter: (tabulka.row_number = 1)
3	[...] Rows Removed by Filter: 20
4	[...] -> Sort (cost=131559.98..131655.34 rows=38142 width=85) (actual time=5150.806..5151.025 rows=23 loops=1)
5	[...] Sort Key: h.localized_name, (count(*)) DESC
6	[...] Sort Method: quicksort Memory: 28kB
7	[...] -> WindowAgg (cost=121982.70..128657.55 rows=38142 width=85) (actual time=5149.849..5150.525 rows=23 loops=1)
8	[...] -> Sort (cost=121982.70..122078.05 rows=38142 width=77) (actual time=5149.793..5150.009 rows=23 loops=1)
9	[...] Sort Key: h.id, (CASE WHEN (mpd.player_slot < 5) THEN m.radiant_win ELSE (NOT m.radiant_win) END), (count(*)) DESC
10	[...] Sort Method: quicksort Memory: 28kB
11	[...] -> HashAggregate (cost=112882.19..119080.27 rows=38142 width=77) (actual time=5149.035..5149.512 rows=23 loops=1)
12	[...] Group Key: h.id, CASE WHEN (round((((au."time")::numeric / (m.duration)::numeric) * '100'::numeric), 5) < '10'::numeric) THEN '0-9'::text WHEN (roun...
13	[...] -> Nested Loop (cost=17431.85..112405.42 rows=38142 width=69) (actual time=3413.926..4726.657 rows=37068 loops=1)
14	[...] -> Index Scan using abilities_pk on abilities a (cost=0.28..8.29 rows=1 width=26) (actual time=0.020..0.042 rows=1 loops=1)
15	[...] Index Cond: (id = 5004)
16	[...] -> Gather (cost=17431.57..106199.05 rows=38142 width=31) (actual time=3413.803..3818.148 rows=37068 loops=1)
17	[...] Workers Planned: 4
18	[...] Workers Launched: 4
19	[...] -> Hash Left Join (cost=16431.57..101384.85 rows=9536 width=31) (actual time=3429.016..4174.986 rows=7414 loops=5)
20	[...] Hash Cond: (mpd.hero_id = h.id)
21	[...] -> Hash Left Join (cost=16428.03..101355.34 rows=9536 width=21) (actual time=3426.464..4012.676 rows=7414 loops=5)
22	[...] Hash Cond: (mpd.match_id = m.id)
23	[...] -> Parallel Hash Left Join (cost=14787.03..99689.31 rows=9536 width=20) (actual time=2256.219..2683.277 rows=7414 loops=5)
24	[...] Hash Cond: (au.match_player_detail_id = mpd.id)
25	[...] -> Parallel Seq Scan on ability_upgrades au (cost=0.00..84877.25 rows=9536 width=12) (actual time=0.102..261.463 rows=7414 loops=5)
26	[...] Filter: (ability_id = 5004)
27	[...] Rows Removed by Filter: 1780506
28	[...] -> Parallel Hash (cost=12770.90..12770.90 rows=161290 width=16) (actual time=2254.159..2254.169 rows=100000 loops=5)
29	[...] Buckets: 524288 Batches: 1 Memory Usage: 27648kB
30	[...] -> Parallel Seq Scan on matches_players_details mpd (cost=0.00..12770.90 rows=161290 width=16) (actual time=0.023..1115.334 rows=100000 l...
31	[...] -> Hash (cost=1016.00..1016.00 rows=50000 width=9) (actual time=1169.762..1169.771 rows=50000 loops=5)
32	[...] Buckets: 65536 Batches: 1 Memory Usage: 2661kB
33	[...] -> Seq Scan on matches m (cost=0.00..1016.00 rows=50000 width=9) (actual time=0.050..583.163 rows=50000 loops=5)
34	[...] -> Hash (cost=2.13..2.13 rows=113 width=14) (actual time=2.467..2.476 rows=113 loops=5)
35	[...] Buckets: 1024 Batches: 1 Memory Usage: 14kB
36	[...] -> Seq Scan on heroes h (cost=0.00..2.13 rows=113 width=14) (actual time=0.049..1.223 rows=113 loops=5)
37	Planning Time: 1.212 ms
38	Execution Time: 5153.229 ms

- Vygenerovaný ORM dotaz

```
1  Subquery Scan on anon_1  (cost=131559.98..132132.11 rows=191 width=77) (actual time=5404.495..5405.009 rows=3 loops=1)
2  [...] Filter: (anon_1.row_number = 1)
3  [...] Rows Removed by Filter: 20
4  [...] ->  Sort  (cost=131559.98..131655.34 rows=38142 width=108) (actual time=5404.472..5404.710 rows=23 loops=1)
5  [...] Sort Key: heroes.name, (count(*)) DESC
6  [...] Sort Method: quicksort  Memory: 28kB
7  [...] ->  WindowAgg  (cost=121982.70..128657.55 rows=38142 width=108) (actual time=5403.545..5404.198 rows=23 loops=1)
8  [...] ->  Sort  (cost=121982.70..122078.05 rows=38142 width=100) (actual time=5403.489..5403.703 rows=23 loops=1)
9  [...] Sort Key: heroes.id, (CASE WHEN (matches_players_details.player_slot < 5) THEN matches.radiant_win ELSE (NOT matches.radiant_win) END), (count(*)) DESC
10 [...] Sort Method: quicksort  Memory: 28kB
11 [...] ->  HashAggregate  (cost=112882.19..119080.27 rows=38142 width=100) (actual time=5402.702..5403.179 rows=23 loops=1)
12 [...] Group Key: heroes.id, CASE WHEN (matches_players_details.player_slot < 5) THEN matches.radiant_win ELSE (NOT matches.radiant_win) END, CASE WHEN (round((((
13 [...] ->  Nested Loop  (cost=17431.85..112405.42 rows=38142 width=92) (actual time=3565.618..4951.777 rows=37068 loops=1)
14 [...] ->  Index Scan using abilities_pk on abilities  (cost=0.28..8.29 rows=1 width=26) (actual time=0.024..0.045 rows=1 loops=1)
15 [...] Index Cond: (id = 5004)
16 [...] ->  Gather  (cost=17431.57..106199.05 rows=38142 width=54) (actual time=3565.524..3992.964 rows=37068 loops=1)
17 [...] Workers Planned: 4
18 [...] Workers Launched: 4
19 [...] ->  Hash Join  (cost=16431.57..101384.85 rows=9536 width=54) (actual time=3559.703..4359.571 rows=7414 loops=5)
20 [...] Hash Cond: (matches_players_details.hero_id = heroes.id)
21 [...] ->  Hash Join  (cost=16428.03..101355.34 rows=9536 width=21) (actual time=3556.497..4185.960 rows=7414 loops=5)
22 [...] Hash Cond: (matches_players_details.match_id = matches.id)
23 [...] ->  Parallel Hash Join  (cost=14787.03..99689.31 rows=9536 width=20) (actual time=2289.029..2748.804 rows=7414 loops=5)
24 [...] Hash Cond: (ability_upgrades.match_player_detail_id = matches_players_details.id)
25 [...] ->  Parallel Seq Scan on ability_upgrades  (cost=0.00..84877.25 rows=9536 width=12) (actual time=0.134..282.201 rows=7414 loops=5)
26 [...] Filter: (ability_id = 5004)
27 [...] Rows Removed by Filter: 1780506
28 [...] ->  Parallel Hash  (cost=12770.90..12770.90 rows=161290 width=16) (actual time=2288.191..2288.204 rows=100000 loops=5)
29 [...] Buckets: 524288  Batches: 1  Memory Usage: 27648kB
30 [...] ->  Parallel Seq Scan on matches_players_details  (cost=0.00..12770.90 rows=161290 width=16) (actual time=0.033..1131.275 rows=100000 loops=5)
31 [...] ->  Hash  (cost=1016.00..1016.00 rows=50000 width=9) (actual time=1266.972..1266.983 rows=50000 loops=5)
32 [...] Buckets: 65536  Batches: 1  Memory Usage: 2661kB
33 [...] ->  Seq Scan on matches  (cost=0.00..1016.00 rows=50000 width=9) (actual time=0.043..629.337 rows=50000 loops=5)
34 [...] ->  Hash  (cost=2.13..2.13 rows=113 width=37) (actual time=3.017..3.027 rows=113 loops=5)
35 [...] Buckets: 1024  Batches: 1  Memory Usage: 16kB
36 [...] ->  Seq Scan on heroes  (cost=0.00..2.13 rows=113 width=37) (actual time=0.033..1.501 rows=113 loops=5)
37 Planning Time: 1.282 ms
38 Execution Time: 5406.786 ms
```

Porovnávané dotazy sú z hľadiska typov operácie rovanké a zhodné sú taktiež polia rows a cost. Líši sa len pole width, ktoré má väčšiu hodnotu pri ORM dotaze. Napríklad v siedmom riadku obsahuje vlastný dotaz width s hodnotou 85,a pri ORM dotaze to je až 108. Taktiež v 36.riadku to je pri vlastnom dotaze 14 a pri ORM 37.

Vo výsledku je tak čas vykonania vlastného dotazu menší zhruba o 300ms.

7. tower_kills

- Vlastný dotaz

1	Sort (cost=192298.92..192298.92 rows=1 width=22) (actual time=99760.288..99761.307 rows=110 loops=1)
2	[...] Sort Key: tabulka2.tower_kills DESC, tabulka2.localized_name
3	[...] Sort Method: quicksort Memory: 33kB
4	[...] -> Subquery Scan on tabulka2 (cost=192298.76..192298.91 rows=1 width=22) (actual time=97846.579..99758.879 rows=110 loops=1)
5	[...] Filter: (tabulka2.row_number = 1)
6	[...] Rows Removed by Filter: 72064
7	[...] -> Sort (cost=192298.76..192298.78 rows=10 width=54) (actual time=97828.646..98763.587 rows=72174 loops=1)
8	[...] Sort Key: tabulka.match_id
9	[...] Sort Method: quicksort Memory: 13214kB
10	[...] -> WindowAgg (cost=192298.36..192298.59 rows=10 width=54) (actual time=94843.583..97056.419 rows=72174 loops=1)
11	[...] -> Sort (cost=192298.36..192298.39 rows=10 width=46) (actual time=94843.525..95546.950 rows=72174 loops=1)
12	[...] Sort Key: tabulka.localized_name, ((count(*) + 1)) DESC
13	[...] Sort Method: quicksort Memory: 10093kB
14	[...] -> GroupAggregate (cost=192297.85..192298.20 rows=10 width=46) (actual time=91260.886..94048.971 rows=72174 loops=1)
15	[...] Group Key: tabulka.match_id, tabulka.id, tabulka.localized_name, tabulka.hero_id, tabulka.lead_hero_id, tabulka.lead_match_id, tabulka.sequence_id
16	[...] -> Sort (cost=192297.85..192297.87 rows=10 width=38) (actual time=91260.831..92241.978 rows=100493 loops=1)
17	[...] Sort Key: tabulka.match_id, tabulka.id, tabulka.localized_name, tabulka.hero_id, tabulka.sequence_id
18	[...] Sort Method: quicksort Memory: 10924kB
19	[...] -> Subquery Scan on tabulka (cost=185187.64..192297.68 rows=10 width=38) (actual time=79688.974..90175.047 rows=100493 loops=1)
20	[...] Filter: ((tabulka.hero_id = tabulka.lead_hero_id) AND (tabulka.match_id = tabulka.lead_match_id))
21	[...] Rows Removed by Filter: 375217
22	[...] -> Sort (cost=185187.64..186203.36 rows=406288 width=42) (actual time=79688.842..84353.967 rows=475710 loops=1)
23	[...] Sort Key: mpd.match_id, go."time", mpd.hero_id
24	[...] Sort Method: quicksort Memory: 58492kB
25	[...] -> WindowAgg (cost=136164.64..147337.56 rows=406288 width=42) (actual time=59627.900..74492.971 rows=475710 loops=1)
26	[...] -> Sort (cost=136164.64..137180.36 rows=406288 width=34) (actual time=59627.822..64257.609 rows=475710 loops=1)
27	[...] Sort Key: mpd.match_id, go."time"
28	[...] Sort Method: quicksort Memory: 49453kB
29	[...] -> WindowAgg (cost=89173.08..98314.56 rows=406288 width=34) (actual time=39797.787..54513.043 rows=475710 loops=1)
30	[...] -> Sort (cost=89173.08..90188.80 rows=406288 width=26) (actual time=39797.708..44492.269 rows=475710 loops=1)
31	[...] Sort Key: mpd.match_id, mpd.hero_id, go."time"
32	[...] Sort Method: quicksort Memory: 49453kB
33	[...] -> Hash Left Join (cost=22411.54..51323.00 rows=406288 width=26) (actual time=10333.658..34695.771 rows=475710 loops=1)
34	[...] Hash Cond: (mpd.hero_id = h.id)
35	[...] -> Hash Left Join (cost=22408.00..50212.96 rows=406288 width=12) (actual time=10331.236..25143.124 rows=475710 loops=1)
36	[...] Hash Cond: (go.match_player_detail_id_1 = mpd.id)
37	[...] -> Seq Scan on game_objectives go (cost=0.00..26738.45 rows=406288 width=8) (actual time=0.041..5119.631 rows=475710 loops=1)
38	[...] Filter: ((match_player_detail_id_1 IS NOT NULL) AND (subtype = 'CHAT_MESSAGE_TOWER_KILL'::text))
39	[...] Rows Removed by Filter: 697686
40	[...] -> Hash (cost=16158.00..16158.00 rows=500000 width=12) (actual time=10330.502..10330.511 rows=500000 loops=1)
41	[...] Buckets: 524288 Batches: 1 Memory Usage: 25581kB
42	[...] -> Seq Scan on matches_players_details mpd (cost=0.00..16158.00 rows=500000 width=12) (actual time=0.018..5134.649 rows=500000 loops=1)
43	[...] -> Hash (cost=2.13..2.13 rows=113 width=14) (actual time=2.362..2.371 rows=113 loops=1)
44	[...] Buckets: 1024 Batches: 1 Memory Usage: 14kB
45	[...] -> Seq Scan on heroes h (cost=0.00..2.13 rows=113 width=14) (actual time=0.026..1.201 rows=113 loops=1)
46	Planning Time: 0.710 ms
47	Execution Time: 99776.255 ms

- Vygenerovaný ORM dotaz

```
1  Sort  (cost=194330.36..194330.36 rows=1 width=22) (actual time=96485.750..96486.850 rows=110 loops=1)
2  [...] Sort Key: anon_1.tower_kills DESC, anon_1.localized_name
3  [...] Sort Method: quicksort  Memory: 33kB
4  [...] ->  Subquery Scan on anon_1  (cost=194330.20..194330.35 rows=1 width=22) (actual time=94922.345..96484.255 rows=110 loops=1)
5  [...] Filter: (anon_1.row_number = 1)
6  [...] Rows Removed by Filter: 72762
7  [...] ->  Sort  (cost=194330.20..194330.22 rows=10 width=54) (actual time=94915.197..95688.317 rows=72872 loops=1)
8  [...] Sort Key: anon_2.match_id
9  [...] Sort Method: quicksort  Memory: 13312kB
10 [...] ->  WindowAgg  (cost=194329.80..194330.03 rows=10 width=54) (actual time=91883.592..94134.678 rows=72872 loops=1)
11 [...] ->  Sort  (cost=194329.80..194329.83 rows=10 width=46) (actual time=91883.511..92599.234 rows=72872 loops=1)
12 [...] Sort Key: anon_2.localized_name, ((count(*) + 1)) DESC
13 [...] Sort Method: quicksort  Memory: 10165kB
14 [...] ->  GroupAggregate  (cost=194329.29..194329.64 rows=10 width=46) (actual time=88303.556..91087.458 rows=72872 loops=1)
15 [...] Group Key: anon_2.match_id, anon_2.id, anon_2.localized_name, anon_2.hero_id, anon_2.lead_hero_id, anon_2.lead_match_id, anon_2.sequence_id
16 [...] ->  Sort  (cost=194329.29..194329.31 rows=10 width=38) (actual time=88303.500..89286.908 rows=101616 loops=1)
17 [...] Sort Key: anon_2.match_id, anon_2.id, anon_2.localized_name, anon_2.hero_id, anon_2.sequence_id
18 [...] Sort Method: quicksort  Memory: 11011kB
19 [...] ->  Subquery Scan on anon_2  (cost=187219.08..194329.12 rows=10 width=38) (actual time=77115.624..87237.219 rows=101616 loops=1)
20 [...] Filter: ((anon_2.hero_id = anon_2.lead_hero_id) AND (anon_2.match_id = anon_2.lead_match_id))
21 [...] Rows Removed by Filter: 374094
22 [...] ->  Sort  (cost=187219.08..188234.80 rows=406288 width=46) (actual time=77115.493..81640.115 rows=475710 loops=1)
23 [...] Sort Key: matches_players_details.match_id, game_objectives."time", heroes.id
24 [...] Sort Method: quicksort  Memory: 58492kB
25 [...] ->  WindowAgg  (cost=138196.08..149369.00 rows=406288 width=46) (actual time=57687.770..72031.452 rows=475710 loops=1)
26 [...] ->  Sort  (cost=138196.08..139211.80 rows=406288 width=42) (actual time=57687.685..62176.211 rows=475710 loops=1)
27 [...] Sort Key: matches_players_details.match_id, game_objectives."time"
28 [...] Sort Method: quicksort  Memory: 52157kB
29 [...] ->  WindowAgg  (cost=89173.08..100346.00 rows=406288 width=42) (actual time=38346.007..52726.742 rows=475710 loops=1)
30 [...] ->  Sort  (cost=89173.08..90188.80 rows=406288 width=30) (actual time=38345.920..42851.048 rows=475710 loops=1)
31 [...] Sort Key: matches_players_details.match_id, heroes.id, game_objectives."time"
32 [...] Sort Method: quicksort  Memory: 49453kB
33 [...] ->  Hash Left Join  (cost=22411.54..51323.00 rows=406288 width=30) (actual time=10018.328..33435.691 rows=475710 loops=1)
34 [...] Hash Cond: (matches_players_details.hero_id = heroes.id)
35 [...] ->  Hash Left Join  (cost=22408.00..50212.96 rows=406288 width=12) (actual time=10016.061..24270.197 rows=475710 loops=1)
36 [...] Hash Cond: (game_objectives.match_player_detail_id_1 = matches_players_details.id)
37 [...] ->  Seq Scan on game_objectives  (cost=0.00..26738.45 rows=406288 width=8) (actual time=0.023..4936.271 rows=475710 loops=1)
38 [...] Filter: ((match_player_detail_id_1 IS NOT NULL) AND (subtype = 'CHAT_MESSAGE_TOWER_KILL'::text))
39 [...] Rows Removed by Filter: 697686
40 [...] ->  Hash  (cost=16158.00..16158.00 rows=500000 width=12) (actual time=10014.547..10014.557 rows=500000 loops=1)
41 [...] Buckets: 524288  Batches: 1  Memory Usage: 25581kB
42 [...] ->  Seq Scan on matches_players_details  (cost=0.00..16158.00 rows=500000 width=12) (actual time=0.034..4962.219 rows=500000 loops=1)
43 [...] ->  Hash  (cost=2.13..2.13 rows=113 width=14) (actual time=2.205..2.214 rows=113 loops=1)
44 [...] Buckets: 1024  Batches: 1  Memory Usage: 14kB
45 [...] ->  Seq Scan on heroes  (cost=0.00..2.13 rows=113 width=14) (actual time=0.027..1.112 rows=113 loops=1)
46 Planning Time: 0.655 ms
47 Execution Time: 96489.616 ms
```

Porovnávané dotazy majú rovnaké typy operácií, aj rovnaký počet riadkov, a líšia sa len v *cost*, ktorý je vyšší pri ORM query. Celkový čas potrebný pre vykonanie je nižší pri ORM dotaze o zhruba 3000ms