

Анализ использования созданной базы данных:

В результате анализа использования БД было выявлено, что наиболее часто используемыми запросами являются:

1. select id из таблицы candidates по полному имени (конкатенация двух полей)
пример запроса из одной из функций:

```
SELECT id
FROM candidates
WHERE CONCAT(first_name, ' ', last_name) = v_member_name;
```

2. update status_id в таблице candidates

```
UPDATE candidates
SET status_id = v_trial_status_id
WHERE id = v_candidate_id;
```

3. select id из таблицы status по полю description
пример запроса из одной из функций:

```
SELECT id INTO v_trial_status_id
FROM status
WHERE description = p_new_statuses[i];
```

4. delete из таблицы trial_in_process по candidate_id и trial_id

```
DELETE
FROM trial_in_process
WHERE candidate_id = v_candidate_id AND trial_id = v_trial_id;
```

5. select id из таблицы trials по полю title

```
SELECT id INTO v_trial_id
FROM trials
WHERE title = v_trial_title;
```

6. update time в таблице trials по id
7. объединение таблиц interaction, interaction_group и
candidate_in_interaction_group по полю interaction_group_id

Созданные индексы:

Созданные индексы:

1. Хеш-индекс по двум полям trial_id и candidate_id для таблицы trials_in_process
2. Хеш-индекс на столбцы first_name и last_name для оптимизации запросов на поиск кандидатов.





3. Хеш-индекс для description в таблице status
4. Хеш-индекс для поля title в таблице trials
5. Хеш-индекс на interaction_group_id и candidate_id для таблицы candidate_in_interaction_group

Были выбраны хеш-индексы, так как все запросы включают в себя проверку только на равенство.

1. **Хеш-индекс на столбцы first_name и last_name** для оптимизации запросов на поиск кандидатов.

```
CREATE INDEX idx_candidates_names_hash ON candidates USING
HASH (first_name, last_name);
```

Статистика данных атрибутов в специальной таблице pg_stats:







| |  n_distinct |  |  correlation |  |
|---|----------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|
| 1 | | -1 | 0.0032287326 | |
| 2 | | -1 | -0.006106794 | |

Отрицательное значение n_distinct означает (если взять по модулю) долю уникальных значений среди всех, причем количество различных значений будет расти при добавлении новых данных в таблицу. Эта доля довольно большая, поэтому созданный индекс с фильтром отсеет большое количество ненужных запросу значений. Но корреляция близка к 0, что делает более дорогостоящим обслуживание индекса. Все же, применение индекса целесообразно за счет большого числа данных в таблице и уникальности значений атрибутов.

2. **Хеш-индекс по полям trial_id и candidate_id** для таблицы trials_in_process

```
CREATE INDEX idx_trial_in_process_hash ON trial_in_process
USING HASH (trial_id, candidate_id);
```

Статистика данных атрибутов в специальной таблице pg_stats:

| |  attname |  |  n_distinct |  |  correlation |  |
|---|---------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|
| 1 | trial_id | | 1 | | 1 | |
| 2 | candidate_id | | -1 | | 0.90666306 | |

Отрицательное значение `n_distinct` для `candidate_id` означает (если взять по модулю) долю уникальных значений среди всех, причем количество различных значений будет расти при добавлении новых данных в таблицу. Эта доля довольно большая, поэтому созданный индекс эффективен. Оба атрибута имеют корреляцию близкую или равную 1, значит обслуживание индекса будет дешевле за счет уменьшения случайного доступа к диску. Применение индекса целесообразно за счет большого числа данных в таблице, уникальности значений атрибутов и их корреляции между физическим порядком строк и логическим порядком значений столбца.

3. Хеш-индекс для `description` в таблице `status`

```
CREATE INDEX idx_status_description_hash ON status USING
HASH (description);
```

В аргументацию применения данного индекса можно привести анализ плана выполнения запроса, который является одним из самых частных. Видно, что СУБД использует индекс для атрибута `description`.

```
explain analyze update candidates set status_id = (SELECT
id from status where description = 'В ПРОЦЕССЕ ИСПЫТАНИЯ')
where status_id = (SELECT id from status where description =
('ДОПУЩЕН К ИСПЫТАНИЮ'));
```

| | QUERY PLAN |
|----|------------------------------------------------------------------------------------------------------------------------------------------|
| 1 | Update on candidates (cost=16.34..359.34 rows=0 width=0) (actual time=60.638..60.639 rows=0 loops=1) |
| 2 | InitPlan 1 (returns \$0) |
| 3 | -> Index Scan using status_description_key on status (cost=0.15..8.17 rows=1 width=4) (actual time=0.002..0.003 rows=1 loops=1) |
| 4 | Index Cond: (description = 'ДОПУЩЕН К ИСПЫТАНИЮ'::text) |
| 5 | InitPlan 2 (returns \$1) |
| 6 | -> Index Scan using status_description_key on status status_1 (cost=0.15..8.17 rows=1 width=4) (actual time=0.002..0.003 rows=1 loops=1) |
| 7 | Index Cond: (description = 'В ПРОЦЕССЕ ИСПЫТАНИЯ'::text) |
| 8 | -> Seq Scan on candidates (cost=0.00..343.00 rows=10000 width=10) (actual time=0.018..1.739 rows=10000 loops=1) |
| 9 | Filter: (status_id = \$1) |
| 10 | Planning Time: 0.164 ms |
| 11 | Trigger for constraint candidates_status_id_fkey: time=24.913 calls=10000 |
| 12 | Execution Time: 85.910 ms |

6. Хеш-индекс для поля `title` в таблице `trials`

```
CREATE INDEX idx_trial_title ON trials USING HASH (title);
```

Разберем частый запрос при создании турнира:

```
explain analyze select id from trials where title in
('{"Dark Continent ExpeditionAYGgfYaqrK6WU9eQ",
                                "Yorknew City
Auction1b1kAceLWMWqD0fLbL",
                                "Zoldyck Family is
Testing GatetdToDle6I5n4PA",
```

```
Expedition6UYWi1IICGvU",
ExterminationfaqhRVudZy09O"}');
"Dark Continent
"Chimera Ant
```

До добавления индекса время исполнения 16.897 мс.

```

1 Seq Scan on trials (cost=0.00..2907.00 rows=5 width=4) (actual time=16.882..16.882 rows=0 loops=1)
2   Filter: ((title)::text = '{"Dark Continent ExpeditionAYGgfYaqRk6WU9eQ",
3     "Yorknew City Auctionlb1kAceLWMWqD0fLbL",
4     "Zoldyck Family is Testing GatetdToDle6I5n4PA",
5     "Dark Continent Expedition6UYWi1IICGvU",
6     "Chimera Ant ExterminationfaqhRVudZy090"}'::text)
7   Rows Removed by Filter: 50000
8 Planning Time: 0.068 ms
9 Execution Time: 16.897 ms

```

После добавления индекса 0.034 мс.

| | QUERY PLAN |
|----|---------------------------------------------------------------------------------------------------------------------|
| 1 | Bitmap Heap Scan on trials (cost=4.04..23.40 rows=5 width=4) (actual time=0.014..0.014 rows=0 loops=1) |
| 2 | Recheck Cond: ((title)::text = '{"Dark Continent ExpeditionAYGgfYaqrK6WU9eQ", |
| 3 | "Yorknew City Auctionlb1kAceLWMWqD0fLbL", |
| 4 | "Zoldyck Family is Testing GatetdToDle6I5n4PA", |
| 5 | "Dark Continent Expedition6UYWi1IIC6vU", |
| 6 | "Chimera Ant ExterminationfaqhRVudZy090"}'::text) |
| 7 | -> Bitmap Index Scan on idx_trial_title (cost=0.00..4.04 rows=5 width=0) (actual time=0.012..0.012 rows=0 loops=... |
| 8 | Index Cond: ((title)::text = '{"Dark Continent ExpeditionAYGgfYaqrK6WU9eQ", |
| 9 | "Yorknew City Auctionlb1kAceLWMWqD0fLbL", |
| 10 | "Zoldyck Family is Testing GatetdToDle6I5n4PA", |
| 11 | "Dark Continent Expedition6UYWi1IIC6vU", |
| 12 | "Chimera Ant ExterminationfaqhRVudZy090"}'::text) |
| 13 | Planning Time: 0.242 ms |
| 14 | Execution Time: 0.034 ms |

7. Хеш-индекс на interaction_group_id и candidate_id для таблицы candidate in interaction group

```
CREATE INDEX idx_interaction_group ON
candidate_in_interaction_group USING HASH
(interaction_group_id);
CREATE INDEX idx_interaction_candidate ON
candidate_in_interaction_group USING HASH (candidate_id);
```

Статистика данных атрибутов в специальной таблице pg_stats:

| | attname ↕ | n_distinct ↕ | correlation ↕ |
|---|----------------------|--------------|----------------|
| 1 | interaction_group_id | -0.15743876 | 1 |
| 2 | candidate_id | 8939 | -0.00032956208 |

Значение `n_distinct` для `candidate_id` означает количество уникальных значений среди всех. Это число довольно большое, поэтому созданный индекс эффективен.

Корреляция для `interaction_group_id` равна 1, значит обслуживание индекса будет дешевле за счет уменьшения случайного доступа к диску.

Применение индексов целесообразно за счет большого числа данных в таблице, уникальности значений атрибута или корреляции между физическим порядком строк и логическим порядком значений столбца.

Также рассмотрим частый запрос на получение данных о взаимодействии для вывода в человекоподобном виде в дальнейшем:

```
explain analyze SELECT
  i.time_start,
  i.time_end,
  ig.description AS interaction_group_description,
  CONCAT(c.first_name, ' ', c.last_name) AS candidate_name
FROM
  interaction i
JOIN
  interaction_group ig ON i.interaction_group_id = ig.id
JOIN
  candidate_in_interaction_group cig ON ig.id =
cig.interaction_group_id
JOIN
  candidates c ON cig.candidate_id = c.id
WHERE
  ig.id = 1;
```

Время выполнения запроса 19.301 мс.

| QUERY PLAN | |
|------------|-------------------------------------------------------------------------------------------------------------------|
| 1 | Nested Loop (cost=0.57..3014.64 rows=6 width=88) (actual time=0.043..19.265 rows=6 loops=1) |
| 2 | -> Nested Loop (cost=0.29..2964.81 rows=6 width=60) (actual time=0.031..19.219 rows=6 loops=1) |
| 3 | -> Nested Loop (cost=0.29..504.81 rows=1 width=60) (actual time=0.024..3.030 rows=1 loops=1) |
| 4 | -> Seq Scan on interaction i (cost=0.00..496.50 rows=1 width=20) (actual time=0.012..3.013 rows=1 ... |
| 5 | Filter: (interaction_group_id = 1) |
| 6 | Rows Removed by Filter: 24999 |
| 7 | -> Index Scan using interaction_group_pkey on interaction_group ig (cost=0.29..8.30 rows=1 width=4... |
| 8 | Index Cond: (id = 1) |
| 9 | -> Seq Scan on candidate_in_interaction_group cig (cost=0.00..2459.94 rows=6 width=8) (actual time=0.006... |
| 10 | Filter: (interaction_group_id = 1) |
| 11 | Rows Removed by Filter: 137349 |
| 12 | -> Index Scan using candidates_pkey on candidates c (cost=0.29..8.30 rows=1 width=49) (actual time=0.006..0.00... |
| 13 | Index Cond: (id = cig.candidate_id) |
| 14 | Planning Time: 0.940 ms |
| 15 | Execution Time: 19.301 ms |

При добавлении индексов время выполнения сокращается до 2.646 мс.

| | | |
|----|---------------------------------------------------------------------------------------------------------------------|---|
| | ⏏ QUERY PLAN | ⌵ |
| 1 | Nested Loop (cost=4.62..581.21 rows=6 width=88) (actual time=0.048..2.593 rows=6 loops=1) | |
| 2 | -> Nested Loop (cost=4.33..531.38 rows=6 width=60) (actual time=0.036..2.556 rows=6 loops=1) | |
| 3 | -> Nested Loop (cost=0.29..504.81 rows=1 width=60) (actual time=0.016..2.534 rows=1 loops=1) | |
| 4 | -> Seq Scan on interaction i (cost=0.00..496.50 rows=1 width=20) (actual time=0.008..2.525 rows=1 lo... | |
| 5 | Filter: (interaction_group_id = 1) | |
| 6 | Rows Removed by Filter: 24999 | |
| 7 | -> Index Scan using interaction_group_pkey on interaction_group ig (cost=0.29..8.30 rows=1 width=44)... | |
| 8 | Index Cond: (id = 1) | |
| 9 | -> Bitmap Heap Scan on candidate_in_interaction_group cig (cost=4.05..26.50 rows=6 width=8) (actual time=0... | |
| 10 | Recheck Cond: (interaction_group_id = 1) | |
| 11 | Heap Blocks: exact=1 | |
| 12 | -> Bitmap Index Scan on idx_interaction_group (cost=0.00..4.04 rows=6 width=0) (actual time=0.014..0... | |
| 13 | Index Cond: (interaction_group_id = 1) | |
| 14 | -> Index Scan using candidates_pkey on candidates c (cost=0.29..8.30 rows=1 width=49) (actual time=0.004..0.005 ... | |
| 15 | Index Cond: (id = cig.candidate_id) | |
| 16 | Planning Time: 0.502 ms | |
| 17 | Execution Time: 2.646 ms | |