



Data modeling with PostgreSQL at the core of Software Development

Boriss Mejías

Holistic System Software Engineer

Air Guitar Player

PgConf Europe – Riga - October 22nd, 2025

Use Case







PostgreSQL Chess Club Tournament



Every one can play as many games as they want,
but only twice against the same player
and only after switching colors of pieces.



```
players = {  
    "player1": {  
        "name": "Judit Polgar",  
        "email": "judit@chess.somewhere"  
    },  
    "player2": {  
        "name": "Ilze Berzina",  
        "email": "ilze@rigachessclub.com"  
    }  
}
```



```
games = [  
    {  
        "white": "Judit Polgar",  
        "black": "Ilze Berzina",  
        "result": "white"  
    }  
]
```



chess.players		
id	int	PK
name	text	UNIQUE
email	text	



```
CREATE SCHEMA chess;
```

```
CREATE TABLE  chess.players(  
    id      INT PRIMARY KEY  
    , name  TEXT  
    , email TEXT  
);
```

```
ALTER TABLE chess.players ALTER COLUMN name SET NOT NULL;
```

```
ALTER TABLE chess.players ADD UNIQUE (name);
```



```
CREATE TABLE chess.players(  
    id    INT PRIMARY KEY  
    , name TEXT NOT NULL UNIQUE  
    , email TEXT  
);
```

```
chess=> INSERT INTO chess.players  
        VALUES (1, 'Judit Polgar', 'judit@chess.somewhere');  
INSERT 0 1
```

```
chess=> INSERT INTO chess.players  
        VALUES (2, 'Ilze Berzina', 'ilze@rigachessclub.com');  
INSERT 0 1
```

```
chess=> INSERT INTO chess.players VALUES (3, 'Judit Polgar');  
ERROR: duplicate key value violates unique constraint "players_name_key"  
DETAIL:  Key (name)=(Judit Polgar) already exists.
```



A constraint is a guarantee



```
CREATE TABLE chess.players(  
    id    INT PRIMARY KEY  
    , name TEXT NOT NULL UNIQUE  
    , email TEXT  
);
```

```
chess=> INSERT INTO chess.players  
        VALUES (1, 'Judit Polgar', 'judit@chess.somewhere');  
INSERT 0 1
```

```
chess=> INSERT INTO chess.players  
        VALUES (2, 'Ilze Berzina', 'ilze@rigachessclub.com');  
INSERT 0 1
```

```
chess=> INSERT INTO chess.players VALUES (3, 'judit polgar');  
INSERT 0 1
```



```
CREATE TABLE chess.players(  
    id    INT PRIMARY KEY  
    , name TEXT NOT NULL UNIQUE  
    , email TEXT  
);
```

```
chess=> ALTER TABLE chess.players DROP CONSTRAINT players_name_key ;  
ALTER TABLE
```

```
chess=> CREATE UNIQUE INDEX unique_player_name  
        ON chess.players (LOWER(name));  
CREATE INDEX
```

```
chess=> INSERT INTO chess.players VALUES (3, 'judit polgar');  
ERROR:  duplicate key value violates unique constraint "unique_player_name"  
DETAIL:  Key (lower(name))=(judit polgar) already exists.
```



Every one can play as many games as they want,
but only twice against the same player
and only after switching colors of pieces.



chess.players		
id	int	PK
name	text	UNIQUE
email	text	

chess.games		
id	int	PK
ts	timestamp	tz
white	int	FK
black	int	FK
result	text	



```
CREATE TABLE chess.games (  
    id      SERIAL PRIMARY KEY  
    , ts    timestamptz  
    , white INT REFERENCES chess.players(id)  
    , black INT REFERENCES chess.players(id)  
    , result text  
);
```



chess.players		
id	int	PK
name	text	UNIQUE
email	text	

chess.games		
id	int	PK
ts	timestamp	tz
white	int	FK
black	int	FK
result	text	



chess.players		
id	int	PK
name	text	UNIQUE
email	text	

chess.games		
id	int	PK
ts	timestamp	tz
white	int	FK
black	int	FK
result	text	

ENUM TYPE

score
white
black
draw
in progress



chess.players		
id	int	PK
name	text	UNIQUE
email	text	

chess.games		
id	int	PK
ts	timestamp	tz
white	int	FK
black	int	FK
result	score	

ENUM TYPE

score
white
black
draw
in progress



chess.players		
id	int	PK
name	text	UNIQUE
email	text	

chess.games		
id	int	PK
ts	timestamp	tz
white	int	FK
black	int	FK
result	score	

ENUM TYPE

score
white
black
draw
in progress



```
CREATE TYPE score AS ENUM (  
    'black'  
    , 'white'  
    , 'draw'  
    , 'in progress'  
);
```

```
CREATE TABLE chess.games (  
    id          SERIAL PRIMARY KEY  
    , ts        timestamptz  
    , white     INT REFERENCES chess.players(id)  
    , black     INT REFERENCES chess.players(id)  
    , result    score  
);
```



Every one can play as many games as they want,
but only twice against the same player
and only after switching colors of pieces.



```
CREATE TABLE chess.games (  
    id      SERIAL PRIMARY KEY  
    , ts    timestamptz  
    , white INT REFERENCES chess.players(id)  
    , black INT REFERENCES chess.players(id)  
    , result score  
);  
  
ALTER TABLE chess.games  
ADD CONSTRAINT limit_games UNIQUE (white, black);
```



```
chess=> INSERT INTO chess.games (ts, white, black, result)
        VALUES ('2025-10-22', 1, 2, 'draw');
INSERT 0 1
```

```
chess=> INSERT INTO chess.games (ts, white, black, result)
        VALUES ('2025-10-22', 1, 2, 'white');
ERROR:  duplicate key value violates unique constraint
"limit_games"
DETAIL:  Key (white, black)=(1, 2) already exists.
```

```
chess=> INSERT INTO chess.games (ts, white, black, result)
        VALUES ('2025-10-22', 2, 1, 'black');
INSERT 0 1
```



We already applied a few
data modeling design principles



chess.players			
id	int	PK	
name	text	UNIQUE	
email	text		

chess.games			
id	int	PK	
ts	timestampz		
white	int	FK	
black	int	FK	
result	score		

ENUM TYPE

score
white
black
draw
in progress

Prevent data duplication



chess.players			
id	int	PK	
name	text	UNIQUE	
email	text		

chess.games			
id	int	PK	
ts	timestampz		
white	int	FK	
black	int	FK	
result	score		

ENUM TYPE

score
white
black
draw
in progress

Each entity must be unique



chess.players			
id	int	PK	
name	text	UNIQUE	
email	text		

chess.games			
id	int	PK	
ts	timestampz		
white	int	FK	
black	int	FK	
result	score		

ENUM TYPE

score
white
black
draw
in progress

A constraint
is not a restriction.
It is a guarantee.



chess.players			
id	int	PK	
name	text	UNIQUE	
email	text		

chess.games			
id	int	PK	
ts	timestampz		
white	int	FK	
black	int	FK	
result	score		

ENUM TYPE

score
white
black
draw
in progress

We don't use business keys
as primary keys



chess.players			
id	int	PK	
name	text	UNIQUE	
email	text		

chess.games			
id	int	PK	
ts	timestampz		
white	int	FK	
black	int	FK	
result	score		

ENUM TYPE

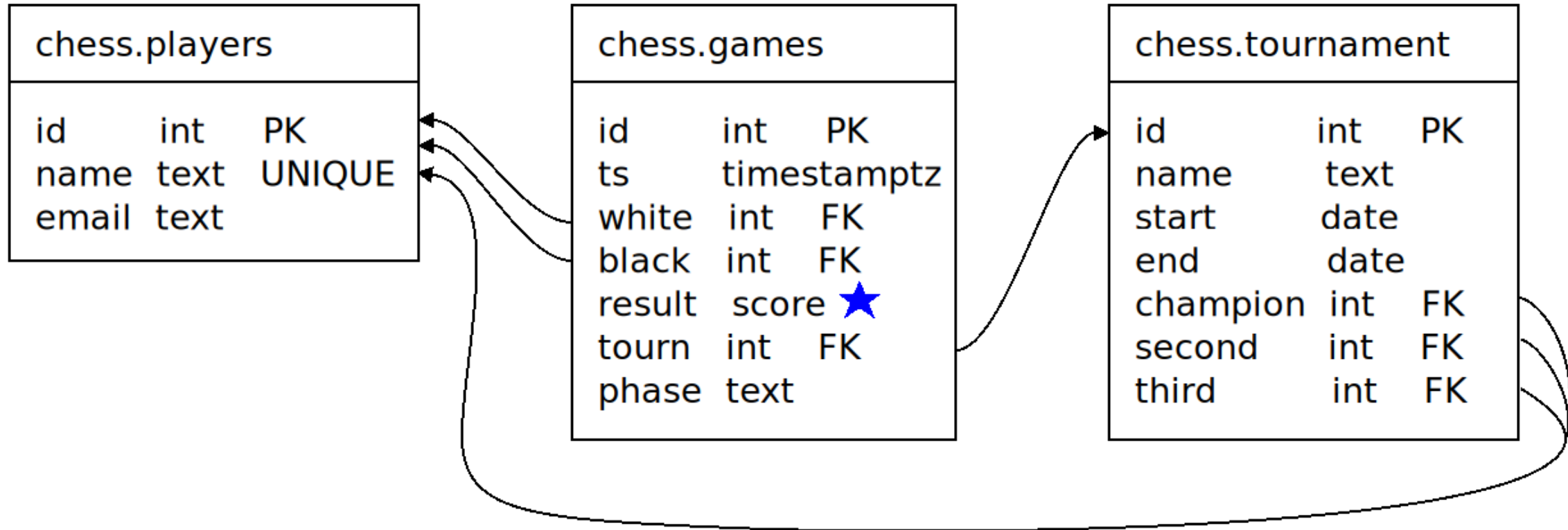
score
white
black
draw
in progress

We don't use natural keys
as primary keys



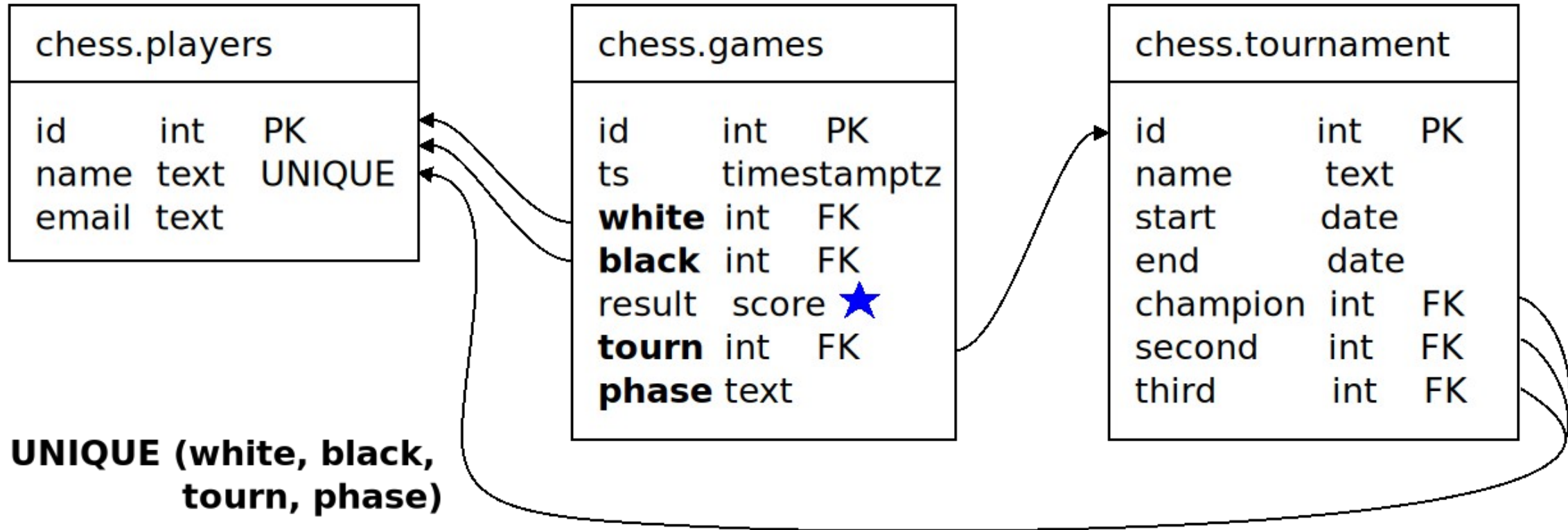
What about the semi-finals?
What about the finals?
What about this year's tournament?





★ score (black, white, draw, in progress)





```
CREATE TABLE chess.tournaments (  
    id          SERIAL PRIMARY KEY  
    , name      TEXT  
    , start_on  DATE  
    , end_on    DATE  
    , champion  INT REFERENCES chess.players(id)  
    , second    INT REFERENCES chess.players(id)  
    , third     INT REFERENCES chess.players(id)  
);
```



```
ALTER TABLE chess.games  
ADD COLUMN tournament INT REFERENCES chess.tournaments(id);
```

```
ALTER TABLE chess.games  
ADD COLUMN phase TEXT;
```

```
ALTER TABLE chess.games  
DROP CONSTRAINT limit_games;
```

```
ALTER TABLE chess.games  
ADD CONSTRAINT limit_games_per_phase_per_tournament  
UNIQUE (white, black, phase, tournament);
```



What about stalemate?



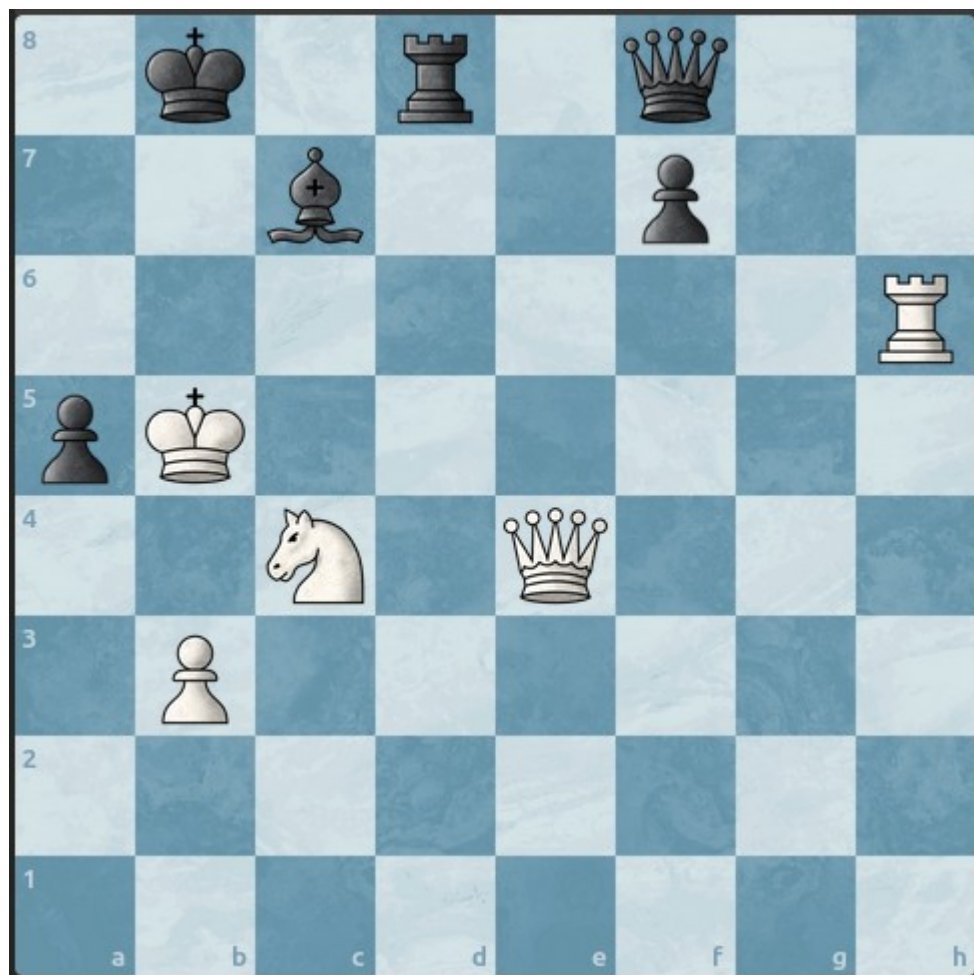
```
chess=> ALTER TYPE score ADD VALUE 'stalemate';
ALTER TYPE
chess=> \dT+
```

List of data types				
Schema	Name	Internal name	Size	Elements
public	score	score	4	black white draw in progress stalemate

(1 row)









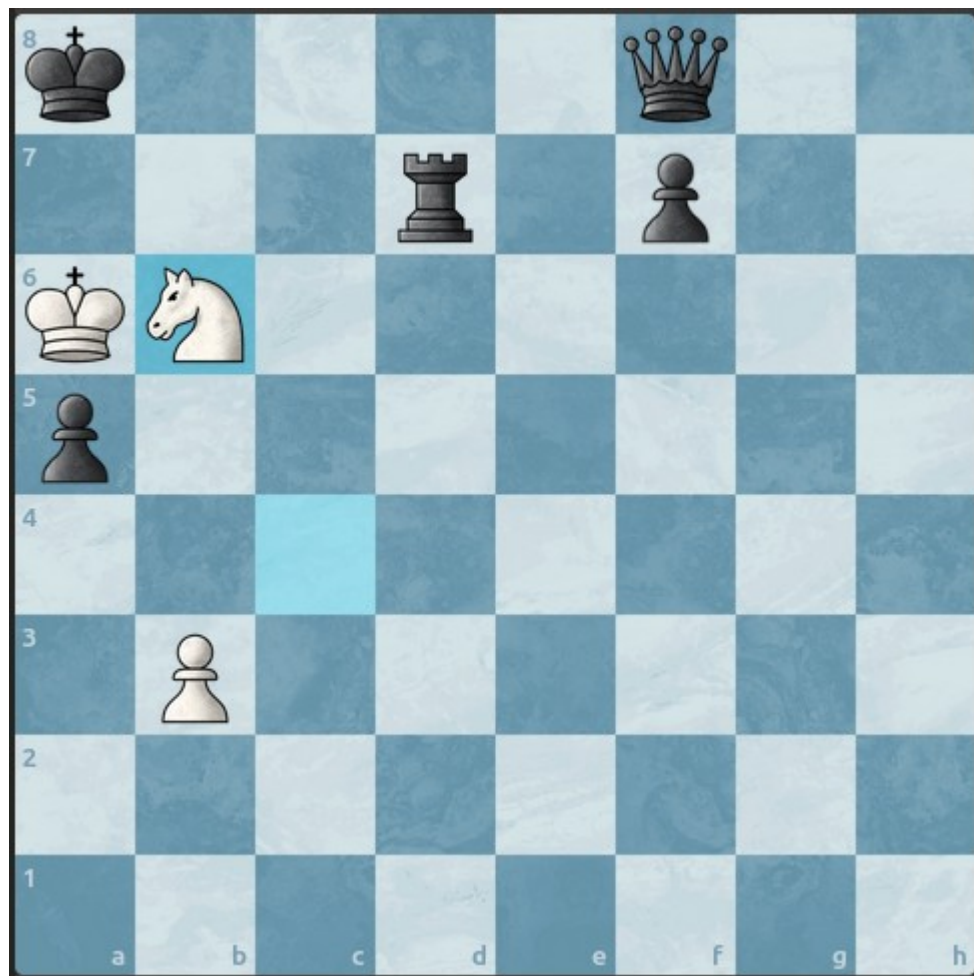


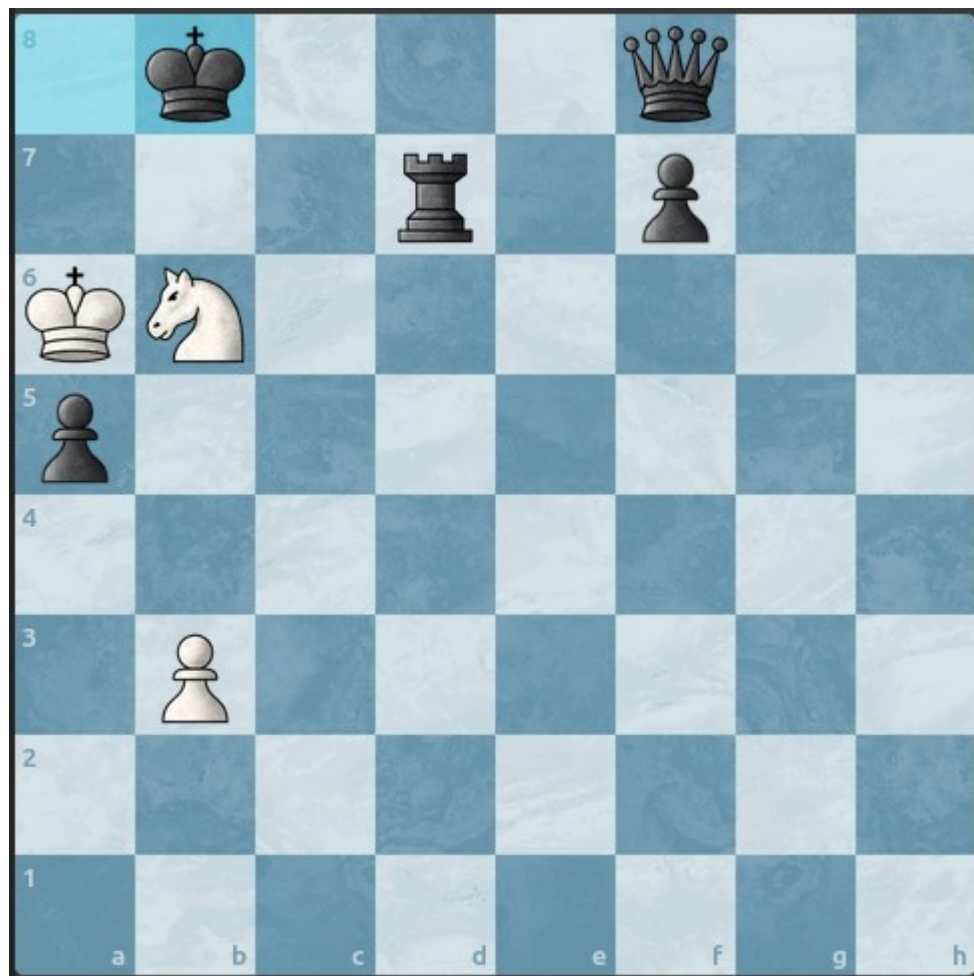


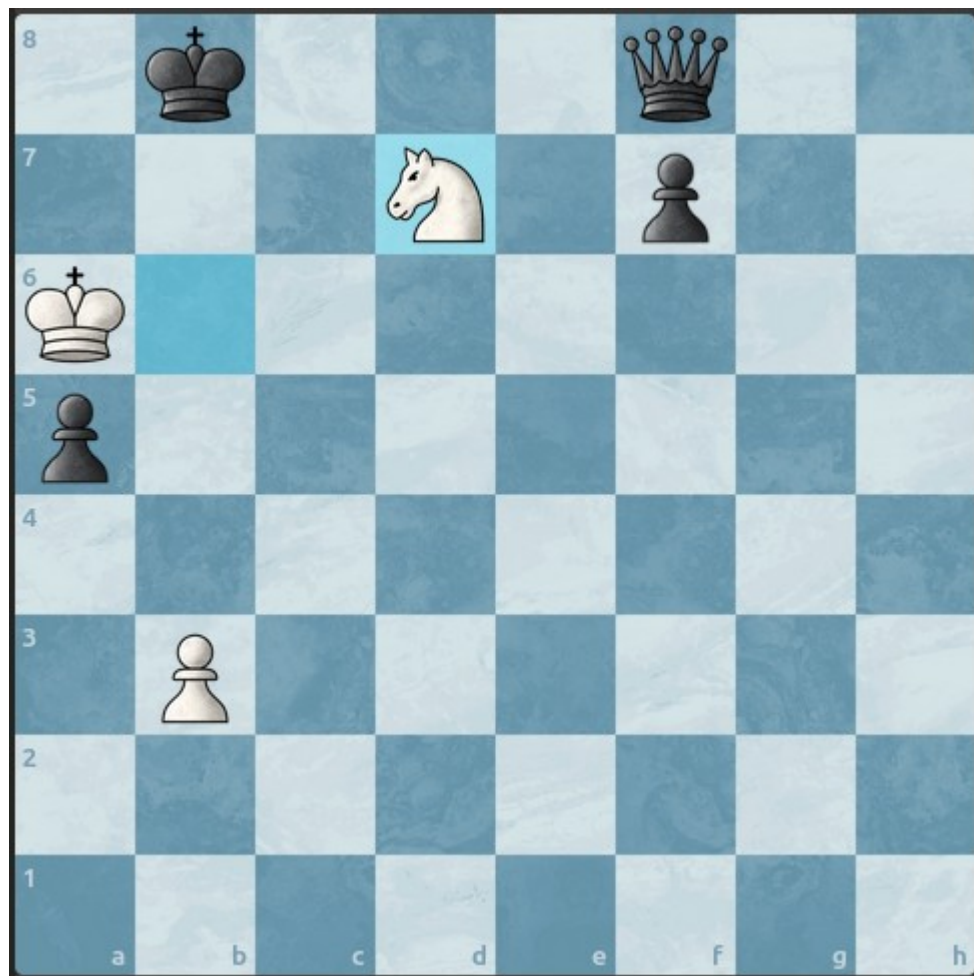


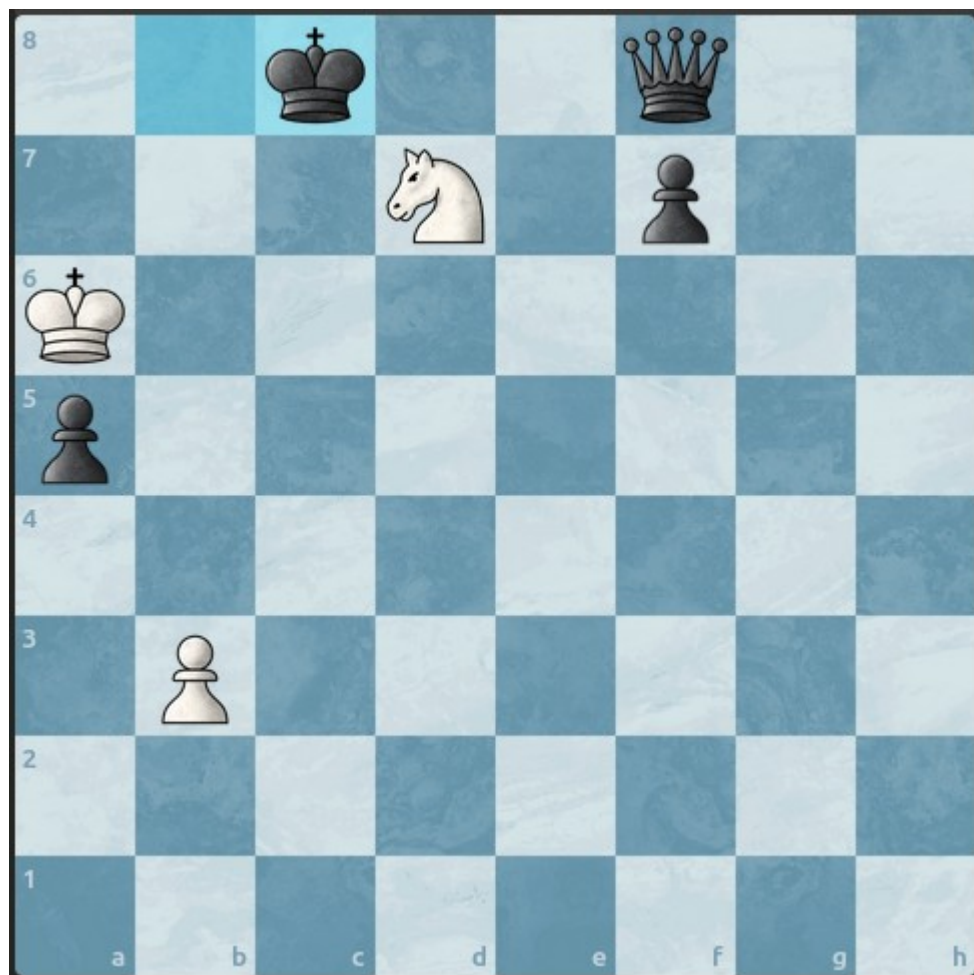


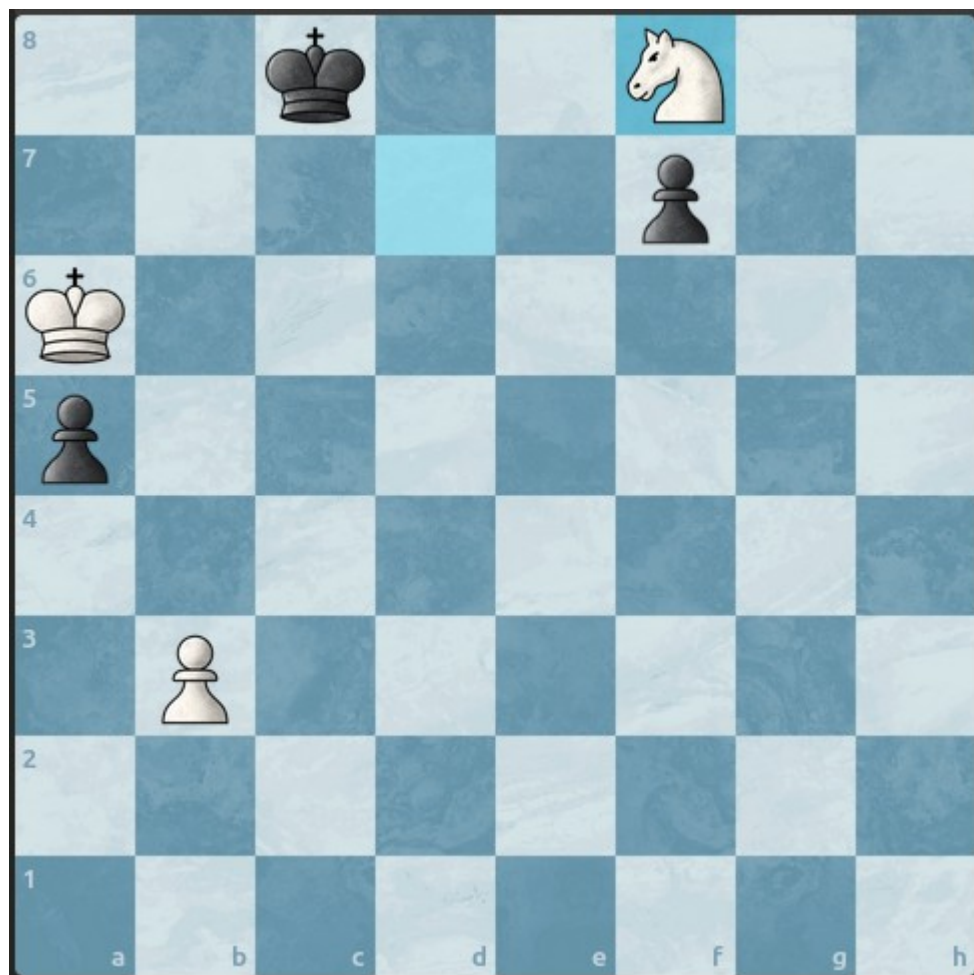












Closing Words



Data Modeling to Implement a Chess Tournament



Data Modeling to Implement Nearly Any Application



Data Modeling to Implement Nearly Any Application

Start with the use case

The Entity Relationship Diagram
should give a lot of information

Apply design principles

Choose the most suitable data type

Postgres gives you a lot of
guarantees and flexibility



Thank You



@tchorix@mastodon.world



[@tchorix.bsky.social](https://bsky.social/@tchorix)



[linkedin.com/in/boriss-mejías-4637401](https://www.linkedin.com/in/boriss-mejías-4637401)



github.com/bmejias

