



ONLINE CINEMA TICKET BOOKING SYSTEM

RYSBEKOV ALMAZBEK

ЦЕЛЬ ПРОЕКТА



Цель проекта:

- Создать базу данных для кинотеатра
- Автоматизировать процесс бронирования билетов
- Исключить ошибки и двойные бронирования



ЧТО ДЕЛАЕТ СИСТЕМА

- 01** регистрировать пользователей
- 02** хранить фильмы и залы
- 03** создавать сеансы
- 04** контролировать свободные места
- 05** бронировать билеты



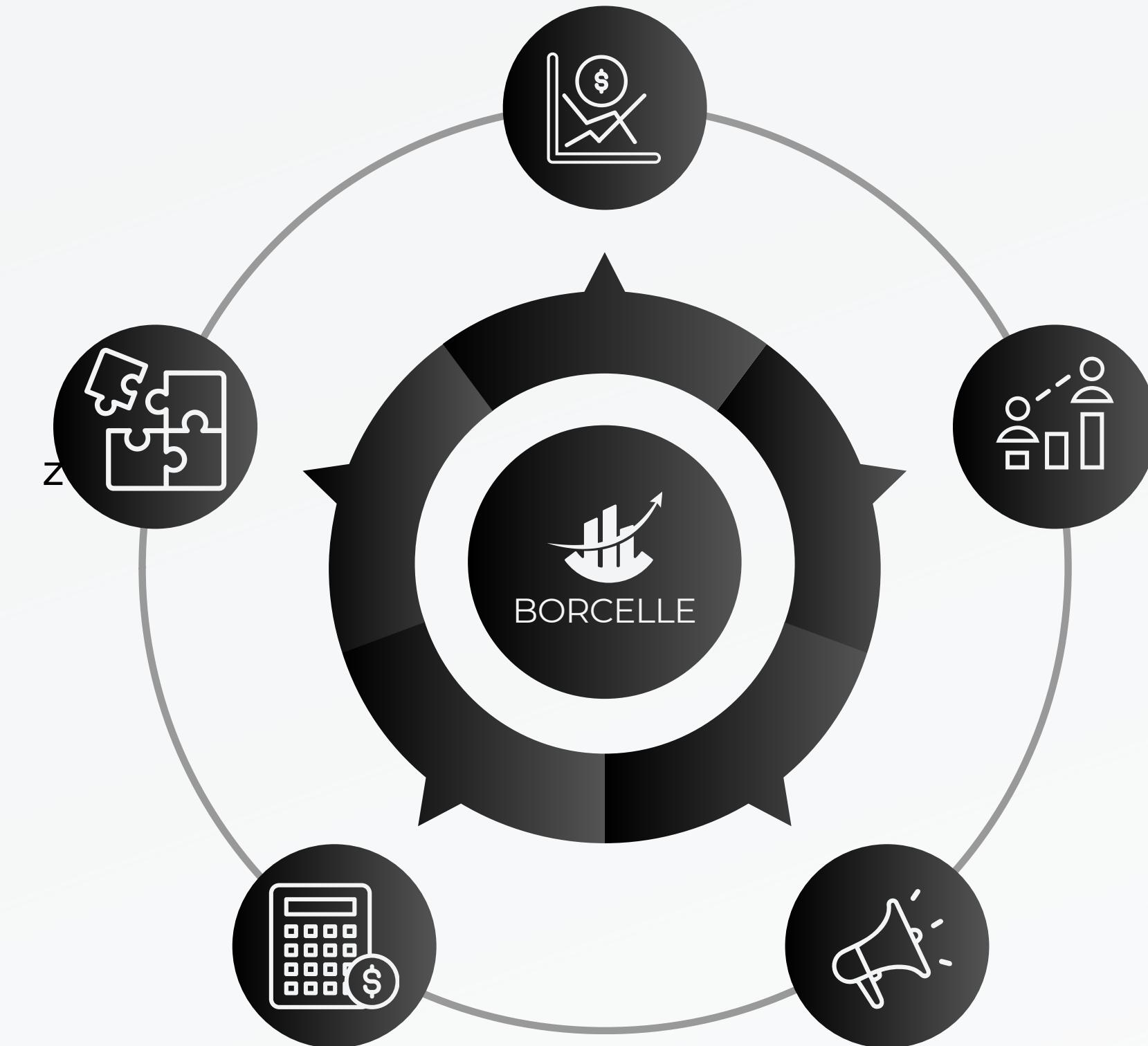
СТРУКТУРА БАЗЫ ДАННЫХ

- 01** **02**
Users **Movies**

- 03** **04**
Halls **Seats**

- 05**
Showtimes

- 06** **07**
Bookings **Tickets**



ОСНОВНЫЕ ТАБЛИЦЫ



MOVIES		
int	movie_id	PK
string	title	
string	genre	
int	duration	
string	rating	
text	description	
date	release_date	
timestamp	created_at	

USERS		
int	user_id	PK
string	name	
string	email	UK
string	phone	
string	password_hash	
timestamp	created_at	

HALLS		
int	hall_id	PK
string	name	
int	capacity	
string	screen_type	
timestamp	created_at	

МЕСТА И СЕАНСЫ БРОНИРОВАНИЯ И БИЛЕТЫ

BOOKINGS		
int	booking_id	PK
int	user_id	FK
int	showtime_id	FK
decimal	total_amount	
string	status	
timestamp	created_at	
timestamp	updated_at	

SEATS		
int	seat_id	PK
int	hall_id	FK
string	row_number	
int	seat_number	
string	seat_type	
timestamp	created_at	

TICKETS		
int	ticket_id	PK
int	booking_id	FK
int	seat_id	FK
decimal	price	
string	status	
timestamp	created_at	

SHOWTIMES		
int	showtime_id	PK
int	movie_id	FK
int	hall_id	FK
time	start_time	
time	end_time	
date	date	
decimal	price	
timestamp	created_at	

ЗАЩИТА ОТ ДВОЙНОГО БРОНИРОВАНИЯ

проверяет,
свободно ли место

```
```sql
CREATE OR REPLACE FUNCTION check_seat_availability()
RETURNS TRIGGER AS $$
DECLARE
 v_showtime_id INTEGER;
 v_seat_taken INTEGER;
BEGIN
 -- Шаг 1: Получаем ID сеанса из бронирования
 SELECT showtime_id INTO v_showtime_id
 FROM bookings
 WHERE booking_id = NEW.booking_id;

 -- Шаг 2: Проверяем, занято ли уже это место на этот сеанс
 SELECT COUNT(*) INTO v_seat_taken
 FROM tickets t
 JOIN bookings b ON t.booking_id = b.booking_id
 WHERE b.showtime_id = v_showtime_id
 AND t.seat_id = NEW.seat_id
 AND t.status = 'active'
 AND b.status != 'cancelled';

```

если занято — выдает ошибку

```
-- Шаг 3: Если место занято — выбрасываем ошибку
IF v_seat_taken > 0 THEN
 RAISE EXCEPTION 'Seat is already booked for this showtime';
END IF;

-- Шаг 4: Если место свободно — разрешаем вставку
RETURN NEW;
END;
$$ language 'plpgsql';
```



## ДОПОЛНИТЕЛЬНАЯ ЗАЩИТА ДАННЫХ

Ограничения (Constraints):  
цена  $\geq 0$   
конец сеанса позже начала  
уникальный email  
уникальные места в зале

# SQL-ЗАПРОСЫ

SELECT

GROUP BY

JOIN

запросы для поиска свободных мест

# ТРАНЗАКЦИИ

создание  
бронирования



создание  
билетов



подтверждение  
покупки



# ИНДЕКСЫ (УСКОРЕНИЕ РАБОТЫ)

поиск бронирований  
пользователя



поиск сеансов по дате

- быстрого поиска данных
- ускорения JOIN
- повышения производительности

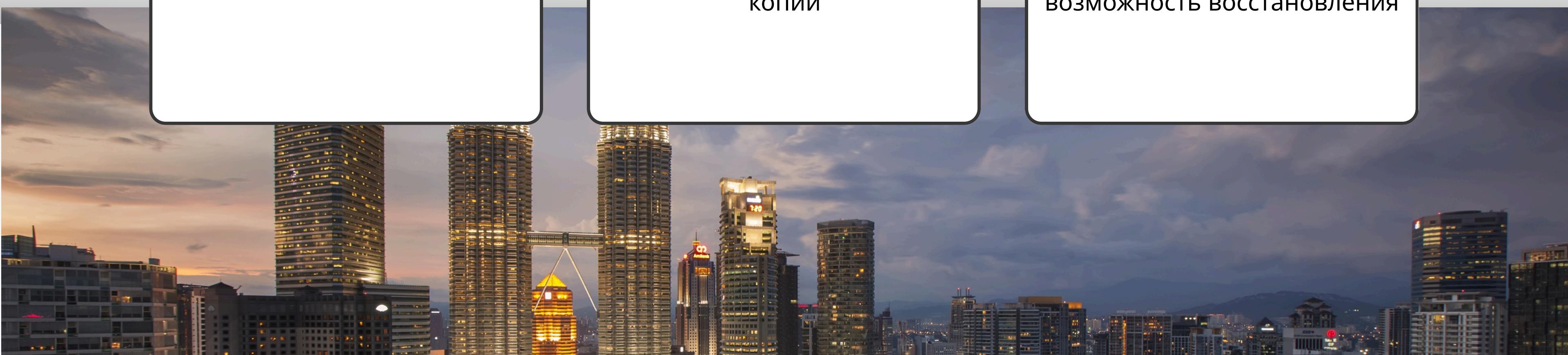


# БАСКИР И ВОССТАНОВЛЕНИЕ

pg\_dump

автоматические резервные  
копии

защита от потери данных  
возможность восстановления



**СПАСИБО ЗА  
ВНИМАНИЕ!  
ГТОВ ОТВЕТИТЬ НА  
ВОПРОСЫ**

