**Федеральное государственное автономное образовательное учреждение высшего образования**

**"Национальный исследовательский университет**

**"Высшая школа экономики"**

**Московский институт электроники и математики им. А.Н.Тихонова**

 Департамент компьютерной инженерии

**Домашняя работа**

по курсу «Базы данных»

Тема: Проектирование базы данных компьютерного клуба

Студенты:

Маденов Денис Александрович

Мосалев Максим Сергеевич

Группа: БИВ225

Преподаватель:

Карпова Ирина Петровна

Москва, 2024 г.

1. **Инфологическое проектирование**

## **1.1** **Анализ предметной области**

База данных создаётся для информационного обслуживания компьютерного клуба: бронирование комнат и игровых станций, закрепление техники за сотрудниками и их графика работы, информация о комнатах и игровых станциях. БД должна содержать данные о сотрудниках, пользователях, игровых станциях, играх и комнатах компьютерного клуба.

В соответствии с предметной областью система строится с учётом следующих особенностей:

* Каждый пользователь может забронировать несколько игровых станций, при этом некоторые игровые станции могут остаться свободными;
* Каждый пользователь может забронировать несколько комнат, при этом некоторые комнаты могут остаться свободными;
* Если пользователь бронирует комнату, то он бронирует все игровые станции в ней;
* Сотрудники клуба имеют доступ к комнатам, при этом каждая комната закреплена за каким-то сотрудником, но у некоторых сотрудников может не быть закрепленной за ним комнаты;
* Среди сотрудников клуба имеются администраторы и техники (инженеры);
* В каждой комнате может иметься несколько игровых станций;
* Несколько игр может быть установлено на одну игровую станцию, при этом на некоторых игровых станциях может не быть игр;
* Компьютерный клуб работает круглосуточно.

Для создания ER-модели выделим сущности предметной области и укажем их атрибуты. **Идентифицирующие** атрибуты выделим полужирным шрифтом, *многозначные* – курсивом, составные подчеркнем.

1. Пользователи

* Имя
* **Электронная почта**
* Телефон
* Дата рождения

1. Сотрудники

* ФИО
* **Паспортные данные**
* **СНИЛС**
* **ИНН**
* Пол
* Дата рождения
* Должность
* Оклад
* *График работы* (дни недели, в которые работает)
* **Логин**

1. Игровые станции

* Название
* Тип (ПК, консоль, VR)
* Цена за 1 час
* Срок введения в эксплуатацию
* Срок вывода из эксплуатации

1. Игры

* Название
* Возрастное ограничение

1. Комната

* Название
* Тип
* Цена за 1 час

Исходя из выявленных сущностей, построим ER–диаграмму (рис. 1).

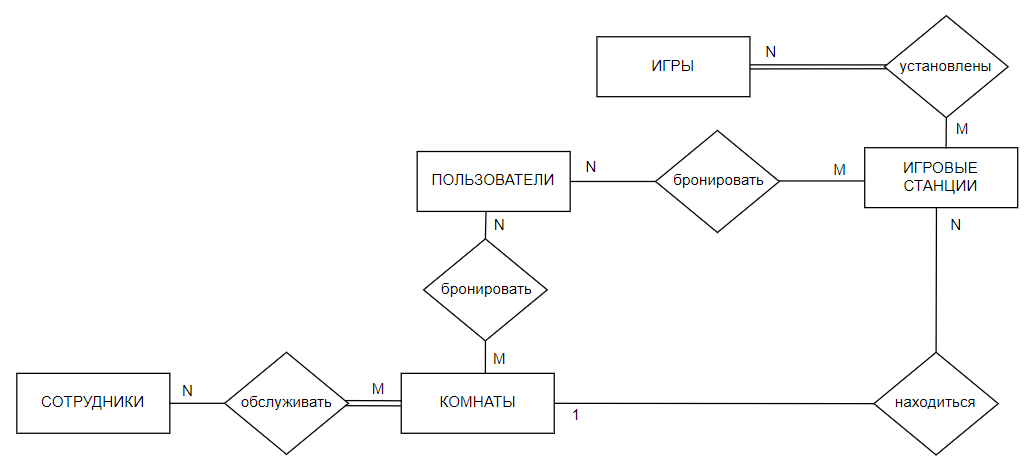


Рисунок 1. ER-диаграмма ПрО «Компьютерный клуб»

## **1.2 Анализ информационных задач и круга пользователей системы**

Определим группы пользователей, их основные задачи и запросы к БД:

1. Пользователь

* Просмотр данных о бронировании комнат и игровых станций
* Просмотр наличия игр на игровых станциях

1. Администратор

* Бронирование комнат
* Бронирование игровых станций
* Заполнение личной информации пользователей

1. Технический специалист

* Изменение информации об игровых станциях и играх

1. Директор клуба

* Закрепление сотрудников за комнатами
* Изменение информации о сотрудниках

# **Логическое проектирование реляционной БД**

## **Преобразование ER–диаграммы в схему базы данных**

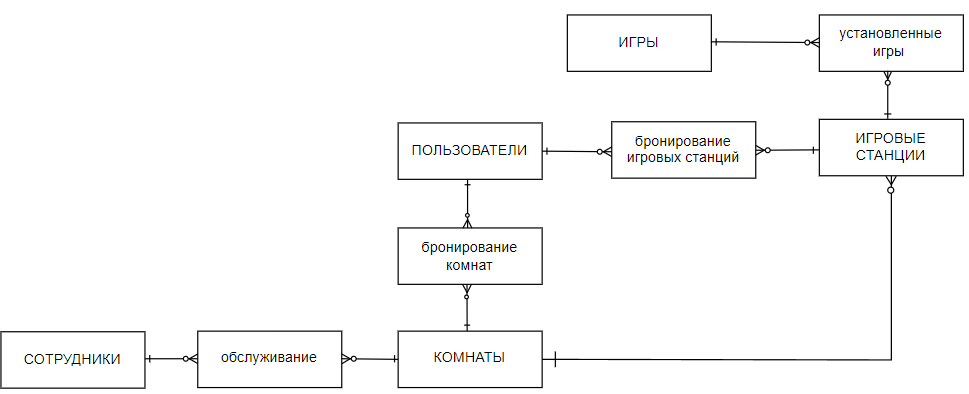


Рисунок 2. Схема базы данных.

## **Составление реляционных отношений**

Для каждого отношения указаны атрибуты с их внутренним названием, типом и длиной. Типы данных обозначаются так: N – числовой, C – символьный тип фиксированной длины, V – символьный тип переменной длины, D – дата.

Таблица 1. Отношение «Пользователи» (Users)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| ID | id | N(4) | Суррогатный первичный ключ |
| Имя | name | V(40) | Обязательное поле |
| Электронная почта | email | V(50) | Обязательное уникальное поле |
| Телефон | phone | N(11) | Обязательное уникальное поле |
| Дата рождения | birth | D | Обязательное поле |

Таблица 2. Отношение «Сотрудники» (Employees)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| ID | id | N(4) | Суррогатный первичный ключ |
| ФИО | name | V(60) | Обязательное поле |
| Паспортные данные | pasp | V(30) | Обязательное уникальное поле |
| СНИЛС | snils | N(11) | Обязательное уникальное поле |
| ИНН | inn | N(12) | Обязательное уникальное поле |
| Пол | sex | C(1) | Обязательное поле, ‘м’ или ‘ж’ |
| Дата рождения | birth | D | Обязательное поле |
| Должность | post | V(30) | Обязательное поле |
| Оклад | salary | N(8,2) | Обязательное поле, >19242 руб. |
| График работы | schedule | V(40) | Обязательное многозначное поле |
| Логин | login | V(30) | Обязательное уникальное поле |

Таблица 3. Отношение «Игры» (Games)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| ID | id | N(4) | Суррогатный первичный ключ |
| Название | name | V(50) | Обязательное уникальное поле |
| Возрастное ограничение | age\_restr | N(2) | Обязательное поле |

Таблица 4. Отношение «Игровые станции» (Stations)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| ID | id | N(4) | Первичный ключ |
| Название | name | V(50) | Обязательное поле |
| Тип | st\_type | V(10) | Обязательное поле |
| Цена за час | cost | N(6) | Обязательное поле, >0 |
| Комната | room\_id | N(4) | Обязательное поле, внешний ключ (к Rooms) |
| Срок введения в экcплуатацию | in\_date | D | Обязательное поле |
| Срок вывода из экcплуатации | out\_date | D | Необязательное поле |

Таблица 5. Отношение «Установленные игры» (Installed\_games)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| ID станции | s\_id | N(4) | Внешний ключ (к Stations) |
| ID игры | g\_id | N(4) | Внешний ключ (к Games) |

Таблица 6. Отношение «Комнаты» (Rooms)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| ID | id | N(4) | Первичный ключ |
| Название | name | V(50) | Обязательное поле |
| Тип | r\_type | V(30) | Обязательное поле |
| Цена за час | cost | N(5) | Обязательное поле, >0 |

Таблица 7. Отношение «Обслуживание комнат» (Service\_rooms)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| ID комнаты | r\_id | N(4) | Внешний ключ (к Rooms) |
| ID сотрудника | e\_id | N(4) | Внешний ключ (к Employees) |
| Начало обслуживания | date­\_start | D | Обязательное поле |
| Конец обслуживания | date\_end | D | Необязательное поле |

Таблица 8. Отношение «Бронирование комнат» (Booking\_room)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| ID комнаты | r\_id | N(4) | Внешний ключ (к Rooms) |
| ID пользователя | u\_id | N(4) | Внешний ключ (к Users) |
| Начало брони | ­b\_start | timestamp | Обязательное поле |
| Конец брони | b\_end | timestamp | Обязательное поле |

Таблица 9. Отношение «Бронирование игровых станций» (Booking\_stations)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| ID игровой станции | st\_id | N(4) | Внешний ключ (к Stations) |
| ID пользователя | u\_id | N(4) | Внешний ключ (к Users) |
| Начало брони | ­b\_start | timestamp | Обязательное поле |
| Конец брони | b\_end | timestamp | Обязательное поле |

## **2.3 Нормализация полученных отношений (до 4НФ)**

Таблица 1 – отношение «Пользователи» останется без изменений.

В таблице 2 – отношение «Сотрудники» – разобьем сложные (составные) атрибуты на простые и вынесем многозначные атрибуты в отдельные отношения. *ФИО* на *Фамилия* и *Имя\_Отчество*. *Паспортные данные* на *Серия и номер*, *Кем выдан*, *Когда выдан*. График работы вынесем в отдельное отношение «График работы» (Schedule). Также имеется транзитивная зависимость: *ID* -> *должность* -> *оклад*, создадим отношение «Должности» (posts), перенесем в него атрибуты *Должность* и *Оклад*, а первичным ключом сделаем название должности.

Таблица 11. Отношение «Сотрудники» (Employees)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| ID | id | N(4) | Суррогатный первичный ключ |
| Фамилия | lname | V(20) | Обязательное поле |
| Имя отчество | fname | V(40) | Обязательное поле |
| Серия и номер паспорта | pasp | C(10) | Обязательное уникальное поле |
| Когда выдан паспорт | p\_date | D | Обязательное поле |
| Кем выдан паспорт | p\_given | V(50) | Обязательное поле |
| СНИЛС | snils | N(11) | Обязательное уникальное поле |
| ИНН | inn | N(12) | Обязательное уникальное поле |
| Пол | sex | C(1) | Обязательное поле, ‘м’ или ‘ж’ |
| Дата рождения | birth | D | Обязательное поле |
| Должность | post | V(50) | Внешний ключ (к Posts) |
| Логин | login | V(30) | Обязательное уникальное поле |

Таблица 12. Отношение «Должности» (posts)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| Должность | post | V(50) | Первичный ключ |
| Оклад | salary | N(8, 2) | Обязательное поле >19242 руб. |

Таблица 13. Отношение «График работы» (Schedule)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| ID сотрудника | e\_id | N(4) | Внешний ключ (к Empoyees) |
| День недели | day\_of\_week | C(2) | Обязательное поле, пн вт ср чт пт сб вс |
| Начало рабочего дня | start\_time | time | Обязательное поле |
| Конец рабочего дня | end\_time | time | Обязательное поле |

В отношении «Игровые станции» атрибут Цена за час зависит только от названия станции и не зависит от её типа (ПК, консоль, VR).

В отношении «Комнаты» есть транзитивная связь *Название*->*Тип*->*Цена за час*. Создадим новое отношение «Типы комнат» (Rooms\_types) и перенесем в него атрибут *Цена за час.*

Таблица 14. Отношение «Комнаты» (Rooms)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| ID | id | N(4) | Первичный ключ |
| Название | name | V(50) | Обязательное поле |
| Тип | r\_type | V(30) | Внешний ключ (к Rooms\_types) |

Таблица 15. Отношение «Типы комнат» (Room\_types)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| Тип | r\_type | V(30) | Первичный ключ |
| Цена за час | cost | N(5) | Обязательное поле, >0 |

Остальные отношения без изменений, так как уже представлены в 4 нормальной форме.

Схема базы данных после нормализации приведена на Рис. 3.

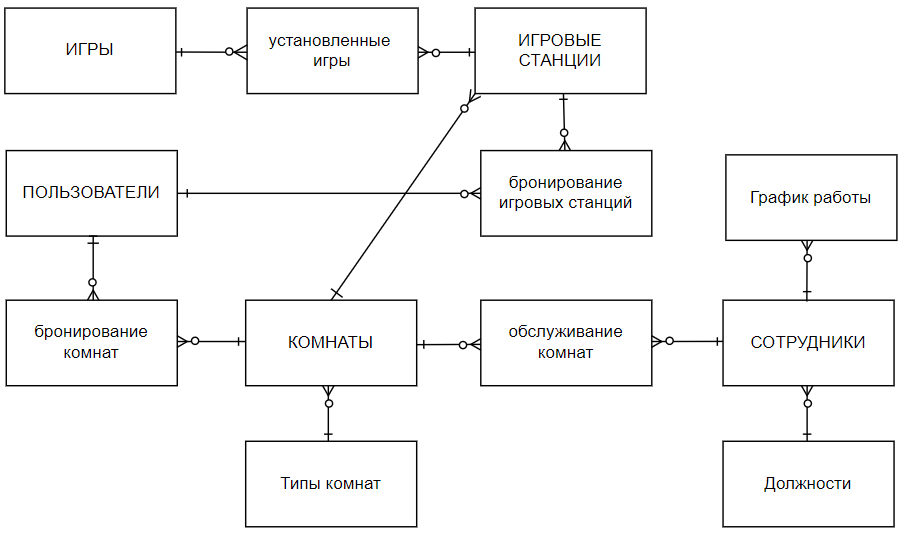


Рисунок 3. Окончательная схема БД компьютерного клуба.

## **2.4 Определение дополнительных ограничений целостности**

Перечислим ограничения целостности, которые не указаны в табл. 1–15.

* 1. Нельзя забронировать игровую станцию, если на это же время забронирована комната, в которой находится эта игровая станция.
  2. Нельзя забронировать комнату, если на это же время забронирована хотя бы одна игровая станция, которая находится в этой комнате.
  3. Нельзя забронировать игровую станцию тем пользователям, у которых возраст меньше возрастного ограничения игры, установленной на данную игровую станцию.
  4. Нельзя забронировать игровую станцию, если она уже забронирована.
  5. Нельзя забронировать комнату, если она уже забронирована.

## **2.5 Описание групп пользователей и прав доступа**

Опишем для каждой группы пользователей права доступа к каждой таблице (Таблица 16). Используются следующие сокращения:

Таблица 16. Обозначения прав доступа

|  |  |
| --- | --- |
| Обозначение | Расшифровка |
| I (insert) | добавление данных |
| S (select) | чтение данных |
| U (update) | модификация данных |
| D (delete) | удаление данных |

Таблица 17. Права доступа к таблицам для групп пользователей

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Таблицы | Группы пользователей (роли) | | | |
| Пользователь | Администратор | Технический специалист | Директор клуба |
| Пользователи |  | SUID |  | S |
| Сотрудники |  |  |  | SUID |
| Должности |  |  |  | SUID |
| График работы |  | S | S | SUID |
| Игровые станции | S | SU | SUID | SU |
| Игры | S | S | SUID | S |
| Установленные игры | S | S | SUID | S |
| Комнаты | S | S | S | SUID |
| Типы комнат | S | SU | S | SUID |
| Обслуживание комнат |  | S | S | SUID |
| Бронирование комнат | SI | SUID | S | S |
| Бронирование игровых станций | SI | SUID | S | S |

## **3. Реализация проекта базы данных**

## **3.1 Создание таблиц**

1. Отношение «Пользователи» (Users)

create table users (

id numeric(4) primary key,

name varchar(40) not null,

email varchar(50) unique not null,

phone numeric(11) unique not null,

birth date not null

);

2. Отношение «Сотрудники» (Employees)

create table employees (

id numeric(4) primary key,

lname varchar(30) not null,

fname varchar(40) not null,

pasp char(10) not null

constraint uniq\_pasp unique,

p\_date date not null,

p\_given varchar(50) not null,

snils numeric(11) not null

constraint uniq\_snils unique,

inn numeric(12) not null

constraint uniq\_inn unique,

sex char(1) not null

constraint check\_sex check(sex in ('м ', 'ж ')),

birth date not null,

post varchar(50) not null

constraint fk\_post references posts,

login varchar(30) not null

constraint uniq\_login unique

);

1. Отношение «Игры» (games)

create table games (

id numeric(4) primary key,

name varchar(50) not null,

age\_restr numeric(2) not null

);

1. Отношение «Игровые станции» (stations)

create table stations (

id numeric(4) primary key,

name varchar(50) not null,

st\_type varchar(10) not null,

cost numeric(6) not null

constraint check\_cost\_st check(cost > 0),

room\_id numeric(4) not null

constraint fk\_room\_id references rooms,

in\_date date not null,

out\_date date

);

1. Отношение «Комнаты» (rooms)

create table rooms (

id numeric(4) primary key,

name varchar(50) not null,

r\_type varchar(30) not null

constraint fk\_r\_type references room\_types

);

1. Отношение «Типы комнат» (room\_types)

create table room\_types (

r\_type varchar(30) primary key,

cost numeric(5) not null

constraint check\_cost\_r check(cost > 0)

);

1. Отношение «Должности» (posts)

create table posts (

post varchar(50) primary key,

salary numeric(8, 2) not null

constraint check\_salary check(salary > 19242)

);

1. Оношение «График работы»с(schedule)

create table schedule (

e\_id numeric(4)

constraint fk\_e\_id references employees,

day\_of\_week char(2) not null

constraint check\_day check(day\_of\_week in

('пн ', 'вт ', 'ср ', 'чт ', 'пт ', 'сб ', 'вс ')),

start\_time time not null,

end\_time time not null

);

1. Отношение «Установленные игры» (installed\_games)

create table installed\_games (

s\_id numeric(4)

constraint fk\_s\_id references stations,

g\_id numeric(4)

constraint fk\_g\_id references games

);

1. Отношение «Обслуживание комнат» (service\_rooms)

create table service\_rooms (

r\_id numeric(4) not null

constraint fk\_r\_id\_serv references rooms,

e\_id numeric(4) not null

constraint fk\_e\_id\_serv references employees,

date\_start date not null,

date\_end date

);

1. Отношение «Бронирование комнат» (booking\_room)

create table booking\_room (

r\_id numeric(4) not null

constraint fk\_r\_id\_bkr references rooms,

u\_id numeric(4) not null

constraint fk\_u\_id\_bkr references users,

b\_start timestamp not null,

b\_end timestamp not null,

constraint check\_b\_time\_room check(b\_end > b\_start)

);

1. Отношение «Бронирование игровых станций» (booking\_stations)

create table booking\_stations (

st\_id numeric(4)

constraint fk\_st\_id\_bkst references stations,

u\_id numeric(4)

constraint fk\_u\_id\_bkst references users,

b\_start timestamp not null,

b\_end timestamp not null,

constraint check\_b\_time\_stations check(b\_end > b\_start)

);

## **3.2 Создание представлений (готовых запросов)**

1. Список свободных комнат на данный момент с указанием типа и ценой

CREATE VIEW free\_rooms AS

SELECT DISTINCT r\_id, name, room\_types.r\_type, cost

FROM booking\_room, room\_types, rooms

WHERE r\_id=id

AND rooms.r\_type=room\_types.r\_type

AND (b\_end <= current\_timestamp or b\_start > current\_timestamp)

AND not exists(select \* from booking\_stations bs, stations s where bs.st\_id = s.id and r\_id = room\_id and bs.b\_start < current\_timestamp and bs.b\_end > current\_timestamp);

2. Список свободных игровых станций на данный момент с указанием комнат, их типа и цены аренды станции

CREATE VIEW free\_stations AS

SELECT DISTINCT s.name, s.st\_type, s.cost, rooms.name room\_name, rooms.r\_type

FROM booking\_stations, room\_types rt, rooms, stations s

WHERE st\_id=s.id

AND s.room\_id=rooms.id

AND rooms.r\_type = rt.r\_type

AND (b\_end <= current\_timestamp or b\_start > current\_timestamp)

AND not exists(select \* from booking\_room br, stations s where br.r\_id = s.room\_id and s.id = st\_id and br.b\_start < current\_timestamp and br.b\_end > current\_timestamp);

3. График работы всех сотрудников

CREATE VIEW schedule\_emp AS

SELECT e.id, e.lname, e.fname, e.post, s.day\_of\_week, s.start\_time, s.end\_time

FROM schedule s, employees e

WHERE e.id=s.e\_id

ORDER BY e.id

4. Список сотрудников, работающих сегодня

CREATE VIEW emp\_today AS

WITH a AS (select extract(isodow from current\_date) a),

b AS(select CASE

WHEN a.a=0 THEN 'вс'

WHEN a.a =1 THEN 'пн'

WHEN a.a =2 THEN 'вт'

WHEN a.a=3 THEN 'ср'

WHEN a.a=4 THEN 'чт'

WHEN a.a=5 THEN 'пт'

WHEN a.a=6 THEN 'сб'

END from a)

SELECT e.id, e.lname, e.fname, e.post

FROM schedule s, employees e

WHERE e.id=s.e\_id and day\_of\_week IN (select \* from b)

ORDER BY e\_id, start\_time

5. Список текущих сотрудников, закреплённых за комнатами

CREATE VIEW emp\_in\_rooms AS

SELECT e.id, lname, fname, post, r.id as room\_id, r.name

FROM rooms r, employees e, service\_rooms s

WHERE s.r\_id=r.id

AND s.e\_id=e.id

AND date\_start <= current\_date

AND date\_end is NULL

6. Упорядоченный список игр, установленных на игровых станциях с указанием комнаты и её типа, в которой находится станция

CREATE VIEW inf\_games AS

SELECT games.name, s.id, s.name as station\_name s.st\_type, rooms.name as room\_name, rooms.r\_type

FROM games, installed\_games ig, rooms, stations s

WHERE ig.g\_id=games.id

AND ig.s\_id=s.id

AND rooms.id=s.room\_id

ORDER BY games.name

7. Список всех пользователей, у которых активен сеанс брони комнаты на данный момент

CREATE VIEW active\_booking\_room AS

SELECT u.id, u.name, u.email, u.phone, r.name room\_name

FROM users u, booking\_room br, rooms r

WHERE u.id=br.u\_id AND r.id=br.r\_id

AND br.b\_start <= current\_timestamp

AND br.b\_end >= current\_timestamp

8. Список всех пользователей, у которых активен сеанс брони игровой станции на данный момент

CREATE VIEW active\_booking\_stations AS

SELECT u.id u\_id, u.name, u.email, u.phone, s.id st\_id, s.name st\_name, r.name room\_name

FROM users u, booking\_stations bs, stations s, rooms r

WHERE u.id=bs.u\_id AND s.id=bs.st\_id AND s.room\_id = r.id

AND bs.b\_start <= current\_timestamp

AND bs.b\_end >= current\_timestamp

9. Данные о текущих станциях с указанием комнат, в которых они находятся и цены брони станции

CREATE VIEW inf\_stations AS

SELECT s.name as station\_name, s.st\_type, s.cost, r.name as room\_name, r.r\_type

FROM rooms r, stations s

WHERE r.id=s.room\_id

10. Информация о комнатах с указанием их цены

CREATE VIEW inf\_rooms AS

SELECT r.name as room\_name, r.r\_type as room\_type, rt.cost

FROM rooms r, room\_types as rt

WHERE r.r\_type=rt.r\_type

Для работы с этими представлениями соответствующим пользователям нужно определить права доступа к представлениям (Таблица 18).

Таблица 18. Права доступа к представлениям

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Представления | Группы пользователей (роли) | | | |
| Пользователь | Администратор | Технический специалист | Директор клуба |
| Свободные комнаты (free\_rooms) | S | S | S |  |
| Свободные станции (free\_stations) | S | S | S |  |
| График работы сотрудников (schedule\_emp) |  | S | S | S |
| Сотрудники, работающие сегодня (emp\_today) |  | S | S | S |
| Сотрудники, закрепленные за комнатами (emp\_in\_rooms) |  | S | S | S |
| Список игр (inf\_games) | S | S | S |  |
| Активная бронь комнат (active\_booking\_room) |  | S | S |  |
| Активная бронь станций (active\_booking\_stations) |  | S | S |  |
| Информация о станциях (inf\_stations) | S | S | S | S |
| Информация о комнатах (inf\_rooms) | S | S | S | S |

## **3.3 Назначение прав доступа**

GRANT SELECT, UPDATE on room\_types to admin;

GRANT ALL on users to admin;

GRANT ALL on employees to director;

…

GRANT SELECT on free\_rooms to user;

GRANT SELECT on free\_rooms to admin;

GRANT SELECT on free\_rooms to tech\_spec;

GRANT SELECT on inf\_stations to PUBLIC;

GRANT SELECT on inf\_rooms to PUBLIC;

…

## **Создание триггеров**

1. Нельзя забронировать игровую станцию, если на это же время забронирована комната, в которой находится эта игровая станция. Реализуем эту проверку с помощью триггера, создав сначала триггерную функцию:

CREATE OR REPLACE FUNCTION check\_room\_booking()

RETURNS TRIGGER AS $$

BEGIN

IF EXISTS (

SELECT 1

FROM booking\_room br

JOIN rooms r ON br.r\_id = r.id

JOIN stations s ON NEW.st\_id = s.id

WHERE br.b\_start < NEW.b\_end

AND br.b\_end > NEW.b\_start

AND r.id = s.room\_id

) THEN

RAISE EXCEPTION 'Нельзя забронировать игровую станцию, так как на это же время забронирована комната, в которой находится данная игровая станция.';

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

После создания триггерной функции можно создать сам триггер:

CREATE TRIGGER prevent\_station\_booking\_overlap

BEFORE INSERT or UPDATE ON booking\_stations

FOR EACH ROW

EXECUTE PROCEDURE check\_room\_booking();

1. Нельзя забронировать комнату, если на это же время забронирована хотя бы одна игровая станция, которая находится в этой комнате.

RETURNS TRIGGER AS $$

BEGIN

IF EXISTS (

SELECT 1

FROM booking\_stations bs

JOIN stations s ON bs.st\_id = s.id

WHERE s.room\_id = NEW.r\_id

AND bs.b\_start < NEW.b\_end

AND bs.b\_end > NEW.b\_start

) THEN

RAISE EXCEPTION 'Нельзя забронировать комнату, так как на это же время забронирована игровая станция, находящаяся в этой комнате.';

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

-- Создаем триггер для обработки операций вставки и обновления

CREATE TRIGGER prevent\_room\_booking\_overlap

BEFORE INSERT OR UPDATE ON booking\_room

FOR EACH ROW

EXECUTE FUNCTION check\_station\_booking();

1. Нельзя забронировать игровую станцию тем пользователям, у которых возраст меньше возрастного ограничения игры, установленной на данную игровую станцию.

CREATE OR REPLACE FUNCTION check\_user\_age\_limit()

RETURNS TRIGGER AS $$

DECLARE

user\_age INT;

game\_age\_restr INT;

BEGIN

-- Получаем возраст пользователя

SELECT EXTRACT(YEAR FROM AGE(users.birth)) INTO user\_age

FROM users

WHERE users.id = NEW.u\_id;

-- Получаем возрастное ограничение игры, установленной на данную игровую станцию

SELECT games.age\_restr INTO game\_age\_restr

FROM games

JOIN installed\_games ON games.id = installed\_games.g\_id

WHERE installed\_games.s\_id = NEW.st\_id;

-- Проверяем, соответствует ли возраст пользователя возрастному ограничению игры

IF user\_age < game\_age\_restr THEN

RAISE EXCEPTION 'Нельзя забронировать игровую станцию. Возраст пользователя меньше возрастного ограничения игры, установленной на эту станцию.';

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

-- Создаем триггер для обработки операций вставки и обновления

CREATE TRIGGER prevent\_user\_age\_limit

BEFORE INSERT OR UPDATE ON booking\_stations

FOR EACH ROW

EXECUTE FUNCTION check\_user\_age\_limit();

1. Нельзя забронировать игровую станцию, если она уже забронирована.

CREATE OR REPLACE FUNCTION check\_station\_availability()

RETURNS TRIGGER AS $$

DECLARE

booked\_count INT;

BEGIN

-- Проверяем количество бронирований для данной игровой станции в указанный период времени

SELECT COUNT(\*) INTO booked\_count

FROM booking\_stations

WHERE st\_id = NEW.st\_id

AND NEW.b\_start < b\_end

AND NEW.b\_end > b\_start;

-- Если количество бронирований больше 0, значит игровая станция уже забронирована

IF booked\_count > 0 THEN

RAISE EXCEPTION 'Нельзя забронировать игровую станцию, так как она уже забронирована.';

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

-- Создаем триггер для обработки операций вставки и обновления

CREATE TRIGGER prevent\_station\_booking\_conflict

BEFORE INSERT OR UPDATE ON booking\_stations

FOR EACH ROW

EXECUTE FUNCTION check\_station\_availability();

1. Нельзя забронировать комнату, если она уже забронирована.

CREATE OR REPLACE FUNCTION check\_room\_availability()

RETURNS TRIGGER AS $$

DECLARE

booked\_count INT;

BEGIN

-- Проверяем количество бронирований для данной комнаты в указанный период времени

SELECT COUNT(\*) INTO booked\_count

FROM booking\_room

WHERE r\_id = NEW.r\_id

AND NEW.b\_start < b\_end

AND NEW.b\_end > b\_start;

-- Если количество бронирований больше 0, значит комната уже забронирована

IF booked\_count > 0 THEN

RAISE EXCEPTION 'Нельзя забронировать комнату, так как она уже забронирована.';

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

-- Создаем триггер для обработки операций вставки и обновления

CREATE TRIGGER prevent\_room\_booking\_conflict

BEFORE INSERT OR UPDATE ON booking\_room

FOR EACH ROW

EXECUTE FUNCTION check\_room\_availability();

## **3.5 Создание индексов**

Анализ готовых запросов показывает, что для повышения эффективности работы с данными необходимо создать индексы для всех внешних ключей. Создадим их:

CREATE INDEX ind\_emp\_post ON employees(post);

CREATE INDEX ind\_st\_r\_id ON stations(room\_id);

CREATE INDEX ind\_rooms\_r\_type ON rooms(r\_type);

CREATE INDEX ind\_sch\_e\_id ON schedule(e\_id);

CREATE INDEX ind\_ins\_games\_s\_id ON installed\_games(s\_id);

CREATE INDEX ind\_ins\_games\_g\_id ON installed\_games(g\_id);

CREATE INDEX ind\_serv\_r\_id ON service\_rooms(r\_id);

CREATE INDEX ind\_serv\_e\_id ON service\_rooms(e\_id);

CREATE INDEX ind\_book\_r\_r\_id ON booking\_room(r\_id);

CREATE INDEX ind\_book\_r\_u\_id ON booking\_room(u\_id);

CREATE INDEX ind\_book\_s\_st\_id ON booking\_stations(st\_id);

CREATE INDEX ind\_book\_s\_u\_id ON booking\_stations(u\_id);

Для ускорения поиска свободных комнат или игровых станций нужен индекс по полям "начало" и "окончание бронирования" в таблицах «Бронирование комнат» и «Бронирование игровых станций».

CREATE INDEX ind\_book\_r ON booking\_room(b\_start, b\_end);

CREATE INDEX ind\_book\_s ON booking\_room(b\_start, b\_end);

## **3.6 Разработка стратегии резервного копирования**

Интенсивность обновления для отношений, связанных с бронированием, достаточно высокая, для остальной БД низкая, поэтому для обеспечения сохранности достаточно проводить полное резервное копирование БД раз в день (ночью), а резервное копирование отношений «Бронирование комнат» и «Бронирование игровых станций» раз в час. Для разработанной БД есть необходимость держать сервер включенным круглосуточно, так как клуб работает круглосуточно, и пользователь может забронировать комнату или игровую станцию в любое время.