

# Схема базы данных сервиса PlayNext (логическая и физическая модель)

## 1 Логическая модель

Логическая модель включает следующие сущности:

- **User** — пользователь (игрок, разработчик, менеджер платформы).
- **Role** — роль пользователя.
- **Preference** — игровые предпочтения игрока (жанры, любимые игры).
- **Game** — игра, добавленная разработчиком.
- **GameMetadata** — описание, жанр, системные требования, целевая аудитория.
- **Rating** — числовая оценка игры по 5-балльной шкале.
- **Review** — текстовый отзыв о игре.
- **Interaction** — событие взаимодействия (просмотр, клик, время в игре).
- **ModerationTask** — задача модерации контента.
- **Recommendation** — персонализированная рекомендация игры игроку.
- **Notification** — уведомление о новой релевантной игре.
- **Report** — аналитический отчёт.
- **SystemMetric** — метрика состояния системы.
- **AuthToken** — токен авторизации для сессии пользователя.

## Бизнес-правила регистрации и авторизации

### 1. Регистрация пользователя

- Новый пользователь создаёт аккаунт с уникальным `login` и `email`.
- Пароль хранится в виде безопасного хеша (`password_hash`).

- Пользователю присваивается роль (`role`) — `player`, `developer` или `manager`.
- После регистрации игрок может заполнить профиль предпочтений (`preferences`).
- Статус аккаунта: `is_active = TRUE`, `is_verified = FALSE` до подтверждения email.

## 2. Авторизация пользователя

- При входе создаётся уникальный токен (`AuthToken.token`) с временем истечения (`expires_at`).
- Один пользователь может иметь несколько активных токенов для разных устройств.
- Истёкшие или отозванные токены (`revoked = TRUE`) не используются для доступа.
- Последний вход пользователя фиксируется в `users.last_login`.
- Проверка токена позволяет определить права доступа по роли (`role`) и активность пользователя (`is_active`).

## 3. Ограничения безопасности

- Пароли никогда не хранятся в открытом виде.
- Токены должны быть случайными и криптографически стойкими.
- После выхода пользователя токен может быть отозван.
- Для восстановления доступа используется подтверждение email или другие методы двухфакторной аутентификации.

## Связи между сущностями

- Один пользователь имеет одну роль.
- Один игрок — один профиль предпочтений.
- Один разработчик может добавить множество игр.
- Одна игра — один набор метаданных и одна задача модерации.
- Один игрок может поставить "одну оценку" и "множество отзывов" на одну игру.
- Рекомендации связывают игрока и игру с мерой релевантности (`score`).
- Один пользователь может иметь несколько токенов авторизации (`AuthToken`).

## 2 Физическая модель

### 2.1 Таблица roles

Поле	Тип	Ограничения
id	SERIAL	PK
name	VARCHAR(20)	NOT NULL, UNIQUE

Начальное заполнение: ('player'), ('developer'), ('manager').

### 2.2 Таблица users

Поле	Тип	Ограничения
id	SERIAL	PK
login	VARCHAR(50)	NOT NULL, UNIQUE
email	VARCHAR(100)	NOT NULL, UNIQUE
password_hash	TEXT	NOT NULL
role_id	INT	NOT NULL, FK → roles(id)
is_active	BOOLEAN	DEFAULT TRUE
is_verified	BOOLEAN	DEFAULT FALSE
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP
last_login	TIMESTAMP	—

Индексы: idx\_users\_role (role\_id), idx\_users\_email (email).

### 2.3 Таблица auth\_tokens

Поле	Тип	Ограничения
id	SERIAL	PK
user_id	INT	NOT NULL, FK → users(id)
token	TEXT	NOT NULL, UNIQUE
expires_at	TIMESTAMP	NOT NULL
revoked	BOOLEAN	DEFAULT FALSE
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP

Индексы: idx\_auth\_tokens\_user (user\_id).

### 2.4 Таблица preferences

user_id	INT	PK, FK → users(id)
genres	TEXT[]	—
favorite_games	TEXT[]	—
updated_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP

## 2.5 Таблица games

id	SERIAL	PK
title	VARCHAR(200)	NOT NULL
developer_id	INT	NOT NULL, FK → users(id)
status	VARCHAR(20)	NOT NULL, DEFAULT 'pending', CHECK IN ('pending', 'approved')
published_at	TIMESTAMP	—
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP

## 2.6 Таблица game\_metadata

game_id	INT	PK, FK → games(id)
description	TEXT	—
genre	VARCHAR(50)	—
system_requirements	TEXT	—
target_audience	TEXT	—

## 2.7 Таблица moderation\_tasks

game_id	INT	PK, FK → games(id)
moderator_id	INT	FK → users(id) (NULL допускается)
status	VARCHAR(20)	NOT NULL, DEFAULT 'pending', CHECK IN ('pending', 'approved')
decision_comment	TEXT	—
reviewed_at	TIMESTAMP	—

## 2.8 Таблица ratings

id	SERIAL	PK
user_id	INT	NOT NULL, FK → users(id)
game_id	INT	NOT NULL, FK → games(id)
rating	SMALLINT	NOT NULL, CHECK (rating BETWEEN 1 AND 5)
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP

UNIQUE(user\_id, game\_id)

---

## 2.9 Таблица reviews

---

id	SERIAL	PK
user_id	INT	NOT NULL, FK → users(id)
game_id	INT	NOT NULL, FK → games(id)
review_text	TEXT	NOT NULL
is_approved	BOOLEAN	DEFAULT FALSE
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP

---

## 2.10 Таблица interactions

---

id	SERIAL	PK
user_id	INT	NOT NULL, FK → users(id)
game_id	INT	FK → games(id) (может быть NULL)
event_type	VARCHAR(30)	NOT NULL, CHECK IN ('view', 'play_time', 'purchase', 'click')
value	TEXT	—
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP

---

## 2.11 Таблица notifications

---

id	SERIAL	PK
user_id	INT	NOT NULL, FK → users(id)
game_id	INT	FK → games(id) (может быть NULL)
message	TEXT	NOT NULL
is_read	BOOLEAN	DEFAULT FALSE
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP

---

## 2.12 Таблица reports

---

id	SERIAL	PK
report_type	VARCHAR(50)	NOT NULL
game_id	INT	FK → games(id) (может быть NULL)
generated_by	INT	FK → users(id) (может быть NULL)
data	JSONB	—
generated_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP

---

## 2.13 Таблица recommendations

id	SERIAL	PK
user_id	INT	NOT NULL, FK → users(id)
game_id	INT	NOT NULL, FK → games(id)
score	FLOAT	NOT NULL, CHECK (score BETWEEN 0.0 AND 1.0)
generated_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP
UNIQUE(user_id, game_id)		

## 2.14 Таблица system\_metrics

id	SERIAL	PK
metric_name	VARCHAR(50)	NOT NULL
value	FLOAT	NOT NULL
recorded_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP

## 3 Порядок заполнения таблиц

1. **roles** — при развёртывании системы.
2. **users** — при регистрации (игрок/разработчик).
3. **auth\_tokens** — при авторизации (создание сессионного токена).
4. **preferences** — после регистрации игрока.
5. **games** и **game\_metadata** — при добавлении игры.
6. **moderation\_tasks** — сразу после добавления игры (**status** = 'pending').
7. После одобрения (**status** = 'approved'):
  - создаются записи в **recommendations**,
  - отправляются **notifications**.
8. **interactions** — в реальном времени при просмотре/клике.
9. **ratings** — при проставлении оценки (1–5 звёзд).
10. **reviews** — при написании отзыва (может быть несколько на игру).
11. **reports** — после запуска анализа менеджером.
12. **system\_metrics** — постоянно в фоне.