



Uniwersytet Gdański  
Wydział Matematyki, Fizyki i Informatyki  
Instytut Informatyki

# Schemat bazy danych online gry

Vadzim Shyshko

Projekt z przedmiotu bazy danych na kierunku informatyka profil ogólnoakademicki na Uniwersytecie Gdańskim.

Gdańsk  
25 maja 2020

## Spis treści

<b>1</b>	<b>Wprowadzenie</b>	<b>2</b>
<b>2</b>	<b>Opis projektu</b>	<b>2</b>
2.1	Potencjalne grupy użytkowników . . . . .	2
2.2	Wymagania funkcjonalne . . . . .	3
2.3	Wymagania niefunkcjonalne . . . . .	3
2.4	Diagram związków encji . . . . .	4
<b>3</b>	<b>Przykłady realizacji bazy danych</b>	<b>5</b>
3.1	Przykłady zawartości najważniejszych tabel . . . . .	6
3.2	Przykłady kilku zapytań i ich wyników . . . . .	7

# 1 Wprowadzenie

Baza danych przeznaczona jest dla online gier. Gracze generują ogromną ilość danych, które muszą być przetwarzane i przechowywane, a które muszą być dostępne w czasie rzeczywistym - niezależnie od tego, czy są to tablice liderów, wirtualne produkty czy dane ośzukujące”. Oczywiście moja baza danych nie jest najbardziej zaawansowana i kompletna. Ale to dla mnie dopiero początek.

- Tabela - podstawowy obiekt bazy danych
- Wiersz tabeli nazywany jest krotką lub rekordem
- Pole to pojedyncza komórka tabeli
- Model relacyjny bazy danych - sposób przechowywania danych w wielu odrębnych, ale połączonych ze sobą tabelach
- Normalizacja to proces organizowania danych w bazie danych
- Klucz podstawowy - pojedyncze pole o unikalnej wartości, które identyfikuje dany rekord
- Klucz obcy - wykorzystuje się go do powiązania dwóch tabel Zazwyczaj jest on kluczem podstawowym innej tabeli
- Raport to prezentacja wybranych informacji z bazy danych
- Kolumna zawiera dane jednego określonego typu
- Relacje to zależności między tabkami, umożliwiające powiązanie ze sobą
- System zarządzania bazami danych to program komputerowy, który służy do przechowywania i modyfikowania danych, np. Microsoft Access
- Kwerenda to zapytanie umożliwiające wyświetlenie pól i rekordów tabeli według kryterium ustalonego przez użytkownika

## 2 Opis projektu

Ten projekt powstał w czasie, gdy zacząłem myśleć o napisaniu mojej gry online. Na początku myślałem o napisaniu prostej gry offline, w której skomplikowana baza danych nie jest potrzebna, ale teraz postanowiłem, że napiszę swoją grę online w stylu mmo rpg. I to jest mój pierwszy krok.

### 2.1 Potencjalne grupy użytkowników

- Administrator – główny zarządca bazy danych, posiada pełen dostęp do bazy danych

- Użytkownik bazy danych jest osobą fizyczną, który ma dostęp do bazy danych i wykorzystuje usługi systemu informatycznego dla aby uzyskać informacje. Na każdym etapie. rozwój bazy danych (projektowanie, wdrożenie, eksploatacja, modernizacja i rozwój, całkowita reorganizacja) z nim różne kategorie użytkowników są połączone.
- Użytkownik końcowy jest główną kategorią użytkowników, w których zainteresowania tworzone są przez bazę danych. W zależności od specyfiki tworzonej bazy danych, krąg użytkowników końcowych może być różny. Mogą to być przypadkowi użytkownicy, którzy od czasu do czasu zgłaszają się do bazy danych po informacje, a także stali użytkownicy. Przynajmniej użytkownikami mogą być np. klienci firmy przeglądającej katalog produktów lub usług.
- Twórcy i administratorzy gier to osoby lub grupy osób, które są do opracowania wymagań dotyczących bazy danych, jej projektowania, tworzenia, efektywnego użytkowania i konserwacji. W trakcie eksploatacji monitoruje funkcjonowanie systemu informatycznego, zapewnia ochronę przed nieuprawnionym dostępem, kontroluje nadmiarowość, spójność, bezpieczeństwo i niezawodność informacji przechowywanych w bazie danych.

## 2.2 Wymagania funkcjonalne

Ta baza danych będzie przechowywać wszystkie informacje o graczu i jego koncie. Jakie bochatery są tworzone na koncie, szczegółowe informacje o bohaterach, na przykład: ilość hp, mana, doświadczenie i kolor włosów. Każdy z bohaterów posiada inwentarz oraz magazyn z przedmiotami, baza danych przechowuje wszystkie informacje o przedmiotach z inwentarza. W bazie danych znajdzie się również historia elementu, czat, zapis czynności postaci. Baza danych przechowuje również informacje o związkach graczy, klanach i rodzinach. Na przykład: ojciec, matka, syn, itd.

## 2.3 Wymagania niefunkcjonalne

Baza danych zrealizowana jest w wolnodostępnym, otwarto źródłowym systemie zarządzania relacyjnymi bazami danych, PostgreSQL 12.2.

Najpierw chciałem użyć MySQL. Ale używałem PostgreSQL. Omawiany silnik bazodanowy jest dużo bardziej zaawansowanym rozwiązaniem niż MySQL. Co za tym idzie, mniejsza jest również jego popularność, jednak ze względu na ogromne możliwości, technologia ta znajduje również ogromną rzeszę fanów, którzy napędzają jej rozwój tworząc dodatkowe narzędzia i biblioteki. Jest to rozwiązanie typu open source. Twórcy bazy zdecydowali się na implementację pełnego standardu ANSI/ISO SQL.

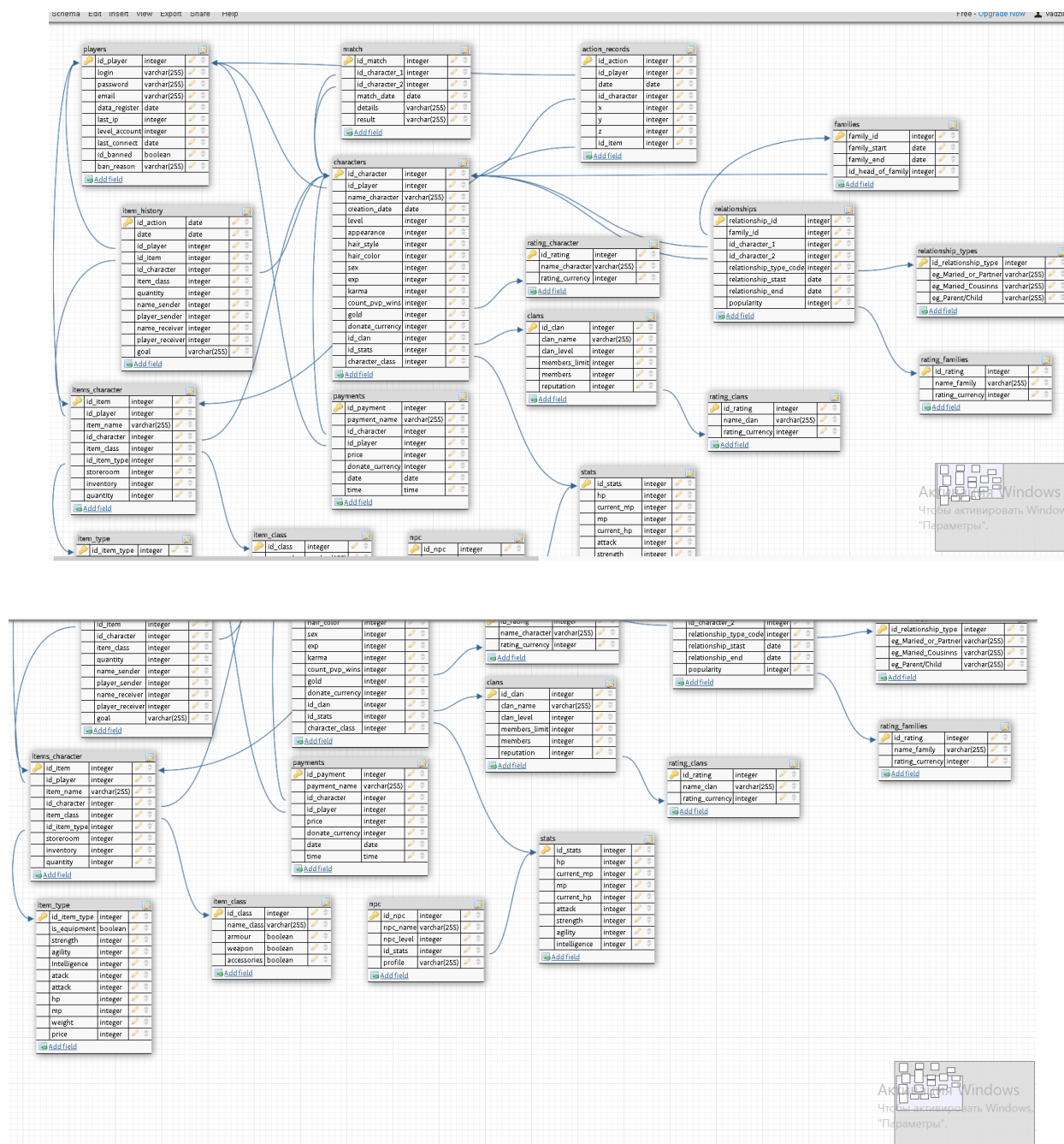
Ogromną zaletą PostgreSQL w porównaniu z innymi rozwiązaniami jest jej “rozszerzalność”. Pozwala ona na tworzenie własnych typów danych czy też dodatkowych funkcji, do pisania których można wykorzystywać składnię innych języków niż SQL.

PostgreSQL świetnie radzi sobie z operacjami równoległymi. W przypadku wielu aplikacji właśnie ta właściwość omawianej bazy jest najbardziej doceniana. Omawiany system posiada również wiele mechanizmów bezpieczeństwa danych. Inną ważną funkcją Postgres’a, o której chciałbym wspomnieć jest wyszukiwanie tekstu (Full Text Search). W

każdym rozwiązaniu bazodanowym znajdziemy funkcję typu LIKE, która pozwala na wyszukiwanie wzorców tekstowych, jednak omawiana baza postawiła na poprawienie tego mechanizmu. Indeksowanie tekstu oraz implementacja funkcji lingwistycznych pozwalają na szybsze i dokładniejsze wyszukiwanie tekstu, szczególnie dobrze sprawdzające się w przypadku języka angielskiego.

Oczywiście nie jest to rozwiązanie idealne dla każdej aplikacji. Mnogość dostępnych funkcji w przypadku mniejszych aplikacji może niepotrzebnie skomplikować, a nawet spowodować wiele procesów, szczególnie tych wymagających szybkiego odczytu danych.

## 2.4 Diagram związków encji



Schemat tego projektu może być podzielony na kilka części. Część w centrum jest związana z bochaterem i kontem, na którym on jest tworzony. Na dole po lewej stronie schematu fragmentu ERD zawierającego informacje o przedmiotach bochaterow . Po prawej stronie schematu znajdują się tabele zawierające informacje o relacjach między przedprzyrodzeniami (rodziny, klany).

### 3 Przykłady realizacji bazy danych

Tutaj opisze tabeli players, characters. Zawierają one kolejno dane użytkownika, informacje o jego bochaterach.

players:

- id\_player - klucz główny tabeli.
- login - pole typu varchar zawierające login użytkownika do 30 zna-ków.
- password - pole typu varchar zawierające hasło użytkownika.
- email - pole typu varchar zawiera adres email użytkownika.
- date\_register - pole typu date zawiera datę rejestracji konta.
- last\_ip - pole typu integer zawiera ip-adres ostatniej sesji.
- level\_account - pole varchar, chciałem zrobić stopni konta(admin, player, tester) ale zapomniałem.
- last\_connect - pole date zawiera datę ostatniej sesji.
- is\_banned - pole boolean czy konto jest zablokowane.
- ban\_reason - pole varchar z przyczyną banu jeżeli konto zablokowane.

characters:

- id\_character - klucz główny tabeli.
- id\_player - klucz obcy tabeli players.
- name\_character - varchar z imieniem bochatera.
- creation\_date - pole date z datą stworzenia bochatera.
- level - int z stopniem bochatera.
- appearance - int z id'im typu wyglądu bochatera.
- hair\_style - pole typu int z typem włosów.
- exp - int z experience'em.

- karma - int z ilością karmy.
- count\_pvp\_wins - int z ilością wygranych pvp.
- gold - pole int z ilością środków ktore ma bochater.
- donate\_currency - pole int z ilością donate środków ktore ma bochater.
- id\_clan - klucz obcy klana.
- id\_stats - klucz obcy charakterystyk.
- character\_class - int z id'im typu bochatera(wizard, warrior).

### 3.1 Przykłady zawartości najważniejszych tabel

Przykładową tabelę sql zamieszczamy przy pomocy polecenia **sqltable**:

tabela players:

Field	Type	Null	Key	Default	Extra
id_player	int	NO	PRI	None	auto_increment
login	varchar(30)	NO		None	
password	varchar(30)	NO		None	
email	varchar(50)	NO		None	
date_register	date	NO		None	
last_ip	int	NO		None	
level_account	int	NO		None	
last_connect	date	NO		None	
is_banned	boolean	NO		None	
ban_reason	varchar(50)	NO		None	

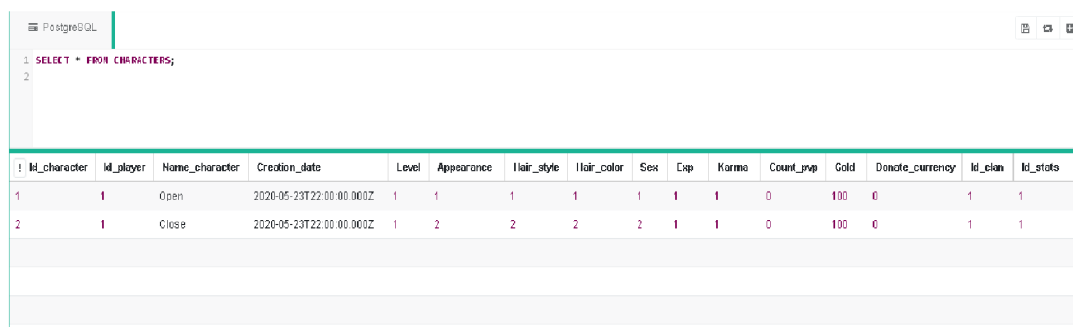
tabela characters:

Field	Type	Null	Key	Default	Extra
id_character	int	NO	PRI	None	auto_increment
id_player	int	NO	MUL	None	
name_character	varchar(30)	NO		None	
creation_date	date	NO		None	
level	int	NO		None	
apearence	int	NO		None	
hair_style	int	NO		None	
exp	int	NO		None	
karma	date	NO		None	
count_pvp_wins	int	NO	MUL	None	

gold	int	NO		None	
donate_currency	int	NO		None	
id_clan	int	NO	MUL	None	
id_stats	int	NO	MUL	None	
character_class	int	NO		None	
+-----+-----+-----+-----+-----+-----+					

## 3.2 Przykłady kilku zapytań i ich wyników

Przykłady umieszczamy przy użyciu specjalnych narzędzi do wstawiania kodu, a dokładniej pakietu **lstlisting**.



The screenshot shows a PostgreSQL query editor with the following query:

```
1 SELECT * FROM CHARACTERS;
2
```

The results are displayed in a table with the following columns:

id_character	id_player	Name_character	Creation_date	Level	Appearance	Hair_style	Hair_color	Sex	Exp	Karma	Count_group	Gold	Donate_currency	id_clan	id_stats
1	1	Open	2020-05-23T22:00:00.000Z	1	1	1	1	1	1	1	0	100	0	1	1
2	1	Close	2020-05-23T22:00:00.000Z	1	2	2	2	2	1	1	0	100	0	1	1

01 | **SELECT \* FROM characters;**

To zapytanie pokazuje wszystkie utworzone osoby (tabela nie jest w tym momencie wypełniona).