Bazy Danych

Projekt zaliczeniowy przedmiotu Bazy Danych

Baza danych sklepu z grami planszowymi

Paweł Hulanicki, 8600

Europejska Uczelnia w Warszawie, informatyka

26 czerwca 2024

Spis treści

1 Wstęp

1.1 Kontekst realistyczny

W związku ze zwiększonym obrotem sklep X z grami planszowymi postanowił zaimplementować bazę danych odpowiadającą za śledzenie liczby dostępnych pozycji, historii zamówień i wypożyczeń oraz danych klientów. Umożliwi to sprawniejsze funkcjonowanie sklepu a w konsekwencji redukcję kosztów operowania. Sklep posiada również wypożyczalnię z grami używanymi.

1.2 Wymagania dotyczące bazy

Zgodnie z wymaganiami baza powinna być napisana zgodnie z systemem zarządzania bazami danych PostgreSQL. Powinna zawierać nie mniej niż 6 encji przy obecności przynajmniej 3 encji nie słownikowych.

1.3 Aplikacja

Wstępny model bazy danych będzie służył wyłącznie do przetestowania i weryfikacji funkcjonalności. W późniejszej fazie do bazy danych zostanie dorobiona aplikacja z GUI (interfejs graficzny) umożliwiająca dodawanie i usuwanie danych dla pracowników, a potem i klientów nieznających języka SQL. W związku z tym przygotowano szereg skryptów które zostaną potem zaimplementowanie w aplikacji.

2 Budowa bazy danych

2.1 Gatunek

Jest to encja zawierająca gatunki gier planszowych w celu łatwejszego dopasowania gier do potrzeb klienta.

kolumna	opis	typ danych	wymagane	klucz?
id_gatunek	unikatowy identyfikator	int	TAK	GŁÓWNY
nazwa_gatunek	nazwa opisująca gatunek	varchar(255)	TAK	

2.2 Autor

Encja zawierająca podstawowe dane autorów gier - imię lub imiona i nazwisko.

kolumna	opis	typ danych	wymagane	klucz?
id_autor	unikatowy identyfikator	int	TAK	GŁÓWNY
imie1	imię autora	varchar(255)	TAK	
imie2	drugie imię autora	varchar(255)	NIE	
nazwisko	nazwisko autora	varchar(255)	TAK	

2.3 Wersja_jezykowa

Encja zawierające dostępne wersje językowe.

kolumna	opis	typ danych	wymagane	klucz?
id_wersja_jezykowa	unikatowy identyfikator	int	TAK	GŁÓWNY
nazwa_wersja	nazwa języka	varchar(255)	TAK	

2.4 Adres_wysylki

Encja zawierająca adresy klientów (jeśli podane) i zrealizowanych zamówień.

kolumna	opis	typ danych	wymagane	klucz?
id_adres	unikatowy identyfikator	int	TAK	GŁÓWNY
adres	adres wysyłkowy zamówień	varchar(255)	TAK	

2.5 Gra_planszowa

Główna część bazy danych, zwiera informacje o dostępnych grach planszowych. Poza podstawowymi informacjami takimi jak tytuł gry, liczba gier na stanie tabela zawiera jeszcze flagę do_wypozyczania, która ma za zadanie przekazać informację czy gra nadaje się do sprzedaży czy też do wykorzystania w wypożyczalni. Gry na sprzedaż nie są udostępniane do wypożyczalni i odwrotnie. W przypadku kilku egzemplarzy tej samej gry do wypożyczenia można dodać je do bazy danych jako osobne wiersze.

kolumna	opis	typ danych	wymagane	klucz?
id_gra_planszowa	unikatowy identyfikator	int	TAK	GŁÓWNY
nazwa_gra	tytuł gry planszowej	varchar(255)	TAK	
gatunek1	nazwa gatunku gry	int	TAK	OBCY
gatunek2	nazwa gatunku gry (jeśli kilka)	int	NIE	OBCY
liczba	liczba dostępnych egzemplarzy	int	NIE	
autor1	dane autora gry	int	TAK	OBCY
autor2	dane drugiego autora	int	NIE	OBCY
ogr_wiekowe	sugerowany minimalny wiek	int	NIE	
wersja_jezykowa	język gry	int	TAK	OBCY
ocena_bgg	ocena wg portalu BGG	decimal	TAK	
cena	cena gry	money	NIE	
do_wypozyczania	jeśli TRUE gra do wypożyczania	boolean	TAK	

2.6 Zakup

Jest to encja przechowująca historię zakupu poszczególnych gier planszowych. Ponieważ przeważająca większość klientów zamawia nie więcej niż jedną grę planszową to nie ma konieczności tworzenia osobnych id dla zamówień - wystarczy osobne id dla każdego aktu sprzedaży danej gry. Przykładowo jak osoba kupi dwa egzemplarze różnej gry to zostaną utworzone dwa wiersze - jeden z jedną grą, drugi z drugą. W przypadku kilku egzemplarzy tej samej gry wystarczy zmodyfikować kolumnę "liczba".

kolumna	opis	typ danych	wymagane	klucz?
id_zakup	unikatowy identyfikator	int	TAK	GŁÓWNY
gra_planszowa	tytuł gry planszowej	int	TAK	OBCY
klient	kupujący	int	TAK	OBCY
data	dzień zrealizowania zamówienia	date	TAK	
adres_wysyłki	adres odbiorcy	int	TAK	OBCY
liczba	liczba egzemplarzy danej gry	int	TAK	

2.7 Wypozyczalnia

Jest to encja przechowująca akty wypożyczania (data rozpoczęcia) i zwrotu (data zakończenia). Dzięki niej można łatwo zidentyfikować osoby które wypożyczyły gry i jeszcze ich nie oddały oraz, w razie zauważonego zniszczenia, łatwo zidentyfikować sprawcę.

kolumna	opis	typ danych	wymagane	klucz?
id_wypozyczalnia	unikatowy identyfikator	int	TAK	GŁÓWNY
gra_planszowa	tytuł gry planszowej	int	TAK	OBCY
klient	kupujący	int	TAK	OBCY
data_wypozyczenia	dzień wypożyczenia gry	date	TAK	
data_zwrotu	dzień zwrotu gry	int		

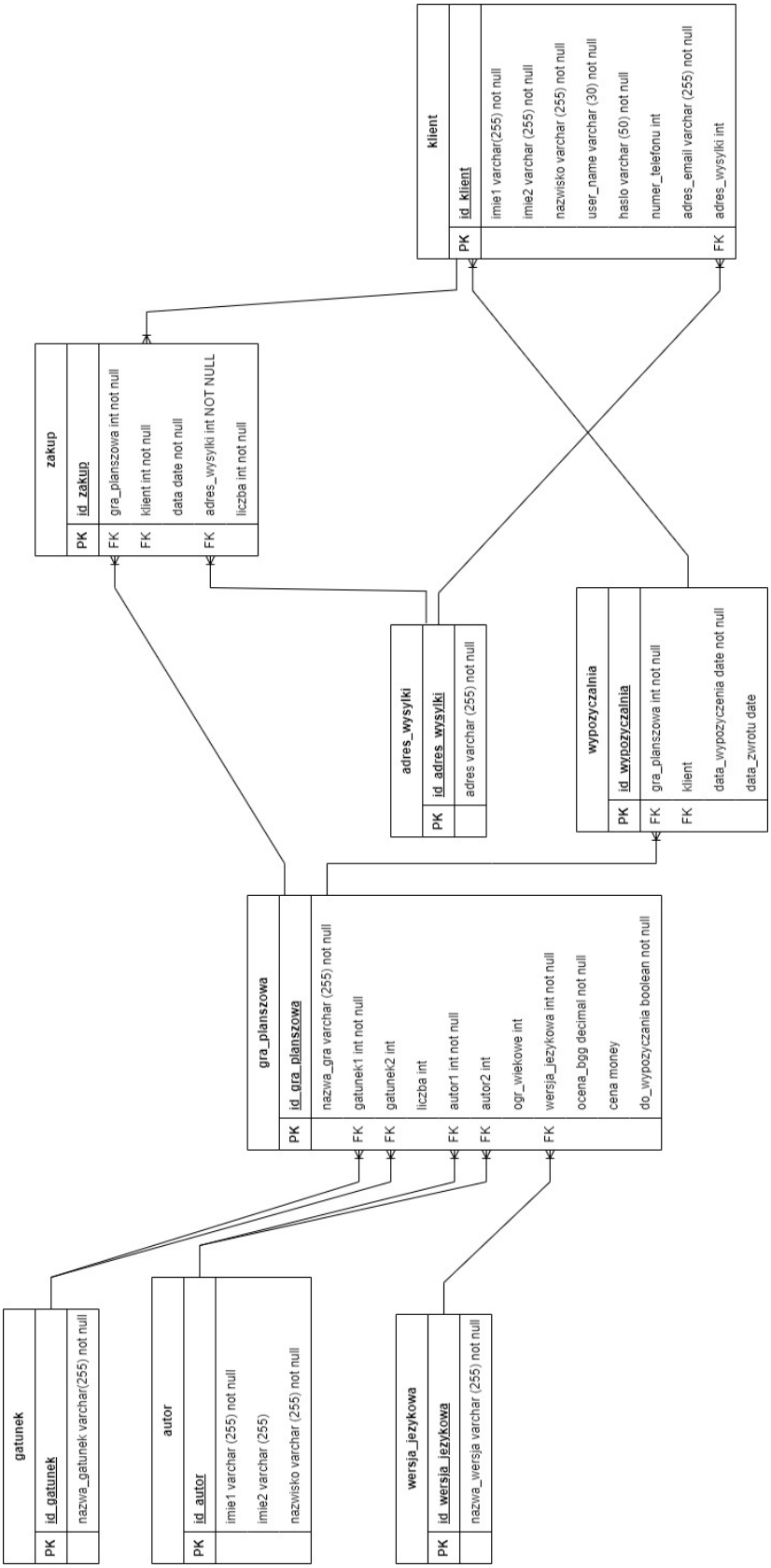
2.8 Klient

Jest to encja przechowujące dane konta klienta. Warto zaznaczyć że to nie w finalnym produkcie dane nie będą przechowywane w ten sposób (to znaczy tabela login nie będzie przechowywać niezmowyfikowanego loginu klienta, tylko np. wynik funkcji hashującej) - wiąże się to z dużym ryzykiem wycieku danych wrażliwych klientów. Jednak obecny model wystarcza na potrzeby stworzenia podstawowego testowania tworzonego systemu. Warto zaznaczyć że adres w encji klient niekoniecznie musi się zgadzać z adresem zamównień złożonych przez klienta.

kolumna	opis	typ danych	wymagane	klucz?
id_klient	unikatowy identyfikator	int	TAK	GŁÓWNY
imie1	imię klienta	varchar(255)	TAK	
imie2	drugie imię klienta	varchar(255)	NIE	
nazwisko	nazwisko klienta	varchar(255)	TAK	
user_name	login klienta do konta	varchar(30)	TAK	
haslo	hasło klienta do konta	varchar(50)	TAK	
numer_telefonu	numer telefonu klienta	int	NIE	
adres_email	adres e-mail klienta	varchar(255)	TAK	
adres_wysyłki	domyślny adres wysyłki	int	NIE	OBCY

2.9 Diagram ERD

Na kolejnej stronie znajduje się diagram ERD opisujący relacje między poszczególnymi encjami wraz z zaznaczonymi kolumnami, typami danych, kluczami głównymi (PK) i kluczami obcymi (FK).



Rys. 1: *Diagram ERD*

3 Utworzenie bazy danych

Poniżej znajduje się kod służący do wygenerowania bazy danych wraz z relacjami przedstawionymi na diagramie ERD. Można do również znaleźć w załączniku.

```
CREATE TABLE gatunek(
        id serial NOT NULL,
        nazwa_gatunek character varying(150) NOT NULL,
        CONSTRAINT id_gatunek PRIMARY KEY (id)
        );

CREATE TABLE autor(
        id serial NOT NULL,
        imie1 character varying(255) NOT NULL,
        imie2 character varying(255),
        nazwisko character varying(255) NOT NULL,
        CONSTRAINT id_autor PRIMARY KEY (id)
        );

Create TABLE wersja_jezykowa(
        id serial NOT NULL,
        wersja character varying(255) NOT NULL,
        CONSTRAINT id_wersja_jezykowa PRIMARY KEY (id)
        );

Create TABLE adres_wysylki(
        id serial NOT NULL,
        adres character varying(255) NOT NULL,
        CONSTRAINT id_adres_wysylki PRIMARY KEY (id)
        );


CREATE TABLE klient(
        id serial NOT NULL,
        imie1 character varying(255) NOT NULL,
        imie2 character varying(255),
        nazwisko character varying(255) NOT NULL,
        user_name character varying(255) NOT NULL,
        haslo character varying(255) NOT NULL,
        numer_telefonu integer,
        adres_email character varying(255) NOT NULL,
        adres_wysylki integer,
        CONSTRAINT id_klient PRIMARY KEY (id),
        CONSTRAINT idf_adres_wysylki FOREIGN KEY (adres_wysylki)
                REFERENCES adres_wysylki (id) MATCH SIMPLE
                ON UPDATE NO ACTION ON DELETE NO ACTION
        );
CREATE TABLE gra_planszowa(
        id serial NOT NULL,
        nazwa_gra character varying(255) NOT NULL,
        gatunek1 integer NOT NULL,
        gatunek2 integer,
        liczba integer,
        autor1 integer NOT NULL,
        autor2 integer,
        ogr_wiekowe integer,
        wersja_jezykowa integer NOT NULL,
```

```
        ocena_bgg decimal NOT NULL,
        cena money,
        do_wypozyczenia boolean NOT NULL,
        CONSTRAINT id_gra_planszowa PRIMARY KEY (id),
          CONSTRAINT idf_gatunek1 FOREIGN KEY (gatunek1)
                REFERENCES gatunek (id) MATCH SIMPLE
                ON UPDATE NO ACTION ON DELETE NO ACTION,
        CONSTRAINT idf_gatunek2 FOREIGN KEY (gatunek2)
                REFERENCES gatunek (id) MATCH SIMPLE
                ON UPDATE NO ACTION ON DELETE NO ACTION,
        CONSTRAINT idf_autor1 FOREIGN KEY (autor1)
                REFERENCES autor (id) MATCH SIMPLE
                ON UPDATE NO ACTION ON DELETE NO ACTION,
        CONSTRAINT idf_autor2 FOREIGN KEY (autor2)
                REFERENCES autor (id) MATCH SIMPLE
                ON UPDATE NO ACTION ON DELETE NO ACTION,
        CONSTRAINT idf_wersja_jezykowa FOREIGN KEY (wersja_jezykowa)
                REFERENCES wersja_jezykowa (id) MATCH SIMPLE
                ON UPDATE NO ACTION ON DELETE NO ACTION

        );
CREATE TABLE zakup(
        id serial NOT NULL,
        gra_planszowa integer NOT NULL,
        klient int NOT NULL,
        data date NOT NULL,
        adres_wysylki integer NOT NULL,
        liczba int NOT NULL,
        CONSTRAINT id_zakup PRIMARY KEY (id),
        CONSTRAINT idf_gra_planszowa FOREIGN KEY (gra_planszowa)
                REFERENCES gra_planszowa (id) MATCH SIMPLE
                ON UPDATE NO ACTION ON DELETE NO ACTION,
        CONSTRAINT idf_klient FOREIGN KEY (klient)
                REFERENCES klient (id) MATCH SIMPLE
                ON UPDATE NO ACTION ON DELETE NO ACTION,
        CONSTRAINT idf_adres_wysylki FOREIGN KEY (adres_wysylki)
                REFERENCES adres_wysylki (id) MATCH SIMPLE
                ON UPDATE NO ACTION ON DELETE NO ACTION
        );
CREATE TABLE wypozyczalnia(
        id serial NOT NULL,
        gra_planszowa integer NOT NULL,
        klient int NOT NULL,
        data_wypozyczenia date NOT NULL,
        data_zwrotu date,
        CONSTRAINT id_wypozyczalnia PRIMARY KEY (id),
        CONSTRAINT idf_gra_planszowa FOREIGN KEY (gra_planszowa)
                REFERENCES gra_planszowa (id) MATCH SIMPLE
                ON UPDATE NO ACTION ON DELETE NO ACTION,
        CONSTRAINT idf_klient FOREIGN KEY (klient)
                REFERENCES klient (id) MATCH SIMPLE
                ON UPDATE NO ACTION ON DELETE NO ACTION
        );
```

4 Skrypty dodające dane

Poniżej znajdują się skrypty pozwalające na dodanie danych do bazy danych. Przykładowe polecenia dodające dane do tabeli są dodatkowo załączone w pliku 'dane.sql'.

4.1 Gatunek

W celu dodania gatunku należy wykonać następujący skrypt (bardzo ważne jest zachowanie wszystkich przecinków i cudzysłowiów jesli są):

```
insert into gatunek(nazwa_gatunek) values('Tu wstaw gatunek');
```

Usuwanie gatunku jest odradzane, ponieważ jest wykorzystywane w tabeli gry_planszowe. Przed usunięciem należy się upewnić że żadna gra nie wykorzystuje id gatunktu który zostanie usunięty. Do usunięcia konieczne jest znalezienie ID gatunku i wykonanie skryptu:

```
delete from nazwa_gatunek where id=ID;
```

Należy oczywiście zastąpić ID liczbą równą znalezionemu identyfikatorowi.

W przypadku konieczności wykonania bardziej zaawansowanej operacji niemożliwej do wykonania na utworzonej w przyszłości aplikacji zaleca się zapoznanie z dokumentacją PostgreSQL

4.2 Autor

Skrypt pozwalający na dodanie autora:

```
insert into autor(imie1,imie2,nazwisko)
values('Tu wstaw imie','Tu drugie imie','Tu nazwisko');
```

Natomiast usunięcie wiersza o danym ID:

```
delete from autor where id=ID;
```

Podobnie jak przy encji gatunek usuwanie autora jest niewzkazane.

4.3 Wersja_jezykowa

Skrypt pozwalający na dodanie wersji językowej ('Polska'):

```
insert into wersja_jezykowa(wersja) values('Tu wstaw wersje jezykowa');
```

Usunięcie wersji językowej o danym ID:

```
delete from wersja_jezykowa where id=ID;
```

Podobnie jak w poprzednich przypadkach usuwanie wersji językowej jest niewzkazane.

4.4 Adres_wysylki

Skrypt pozwalający na dodanie nowego adresu:

```
insert into adres_wysylki(adres) values('Tu wpisac adres');
```

Ponieważ adres jest wykożystywany w zamówieniach usuwanie i modyfikowanie go jest zabronione z wyłączeniem skarnych przypadków, w związku z tym aplikacja nie będzie pozwalać regularnym pracownikom na modyfikację tych danych.

4.5 Gra_planszowa

Dodanie nowej gry planszowej:

```
insert into gra_planszowa(nazwa_gra,gatunek1,gatunek2,liczba,autor1,autor2,
ogr_wiekowe,wersja_jezykowa,ocena_bgg,cena,do_wypozyczenia)
values(
'Tu wstaw tytul',
tu wstaw id gatunku,
tu id drugiego gatunku,
tu liczbe egzemplarzy,
tu id autora,
tu id drugiego autora,
tu ograniczenie wiekowe,
tu id wersji jezykowej,
tu ocene bgg jako liczbe z kropka! (np 6.5, NIE 6,5),
tu wstaw cene (tez z kropka nie z przecinkiem!),
tu wstaw TRUE jesli gra jest do wypozyczenia, FALSE jesli do sprzedazy;
```

Warto zaznaczyć że wstawienie nowej gry wymaga znajomości szeregu id: gatunek, autor, wersja językowa. W przypadku niewystępowania danego gatunku w bazie danych należy go najpierw dodać za pomocą polecenia wzkazanego w paragrafie Gatunek, analogicznie dla pozostałych id. Przykładowo, aby sprawdzić id gatunku 'strategiczna' (o nazwie kolumny z nazwa_strategiczna) należy wykonać polecenie:

```
select id, nazwa_gatunek from gatunek
WHERE nazwa_gatunek = 'strategiczna'
```

W przypadku konieczności znalezienia innych id należy

- zastąpić 'from gatunek' tablelą w której znajduje się pożądzana wielkość, np 'from wersja_jezykowa',
- zastąpić oba wyrażenia nazwa_gatunek odpowiednią nazwą kolumny (nazwy kolumn można znaleźć w rozdziale poświęconym encjom lub na załączonym diagramie ERD),
- w przypadku braku wyniku należy upewnić się że wyrażenie zostało napisane bez żadnych błędów,
- w razie dalszego braku wyników istnieje bardzo duża szansa że poszukiwany autor, wersja językowa lub gatunek nie są uwzględnione w bazie danych. W tej sytuacji należy je dodać za pomocą odpowiednich skryptów opisanych w tym dziale.

Jeśli gra ma dokładnie jednego autora lub jeden gatunek to należy pozostawić odpowiednio miejsca na drugiego autora i drugi gatunek puste (zostawiając przecinek).
Usunięcie gry o znanym ID można zrealizować następującym skryptem:

```
delete from gra_planszowa where id=ID;
```

W przypadku konieczności zmiany danego pola (najczęściej liczba i cena) gry o znanym ID nalezy wykonać następujące polecenie:

```
update gra_planszowa
set liczba = N
where id = ID;
```

Gdzie N - liczba gier po zmianie, ID - identyfikator gry planszowej. W przypadku zmiany ceny należy zastąpić wyrażenie 'set liczba' na 'set cena'. Jeśli konieczne jest dodanie lub odjęcie ceny lub liczby elementów należy wykonać podobny kod:

```
update gra_planszowa
set liczba = liczba + D
where id = ID;
```

Gdzie D to liczba egzeplarzy którą trzeba dodać. W przypadku zmian innych pól należy zwrócić uwagę na nazwę i typ danych kolumny, które można znaleźć w diagramie ERD.

4.6 Klient

W celu dodania nowego klienta należy wykonać następujące polecenie:

```
insert into klient(imie1,imie2,nazwisko,user_name,haslo,
numer_telefonu,adres_email,adres_wysylki)
values(
'Tu wstaw pierwsze imie',
'Tu drugie imie',
'Tu nazwisko',
'tu login',
'tu haslo',
'tu numer telefonu',
'tu adres e-mail',
 tu id_adresu);
```

Docelowo adres będzie automatycznie przypisywany klientowi przez aplikację, w razie konieczności ręcznego dopasowania należy znaleźć id adresu w tabeli adres_wysylki za pomocą skryptu:

```
select id, adres from adres_wysylki
WHERE adres = 'Tu wstaw adres'
```

Usunięcie użytkownika o danym ID jest zabronione regularnym pracownikom, ponieważ dane użytkowników są istotne do śledzenia zamówień i wypożyczalni.

W celu zmiany wartości kolumny dla użytkownika o znanym ID należy wykonać skrypt:

```
update klient
set tu_wstaw_kolumne = 'tu_wstaw_tekst'
where id = ID;
```

4.7 Zakup

W celu dodania wiersza należy wykonać skrypt:

```
insert into zakup(gra_planszowa,klient,data,adres_wysylki,liczba)
values(
tu_wstaw_id_gry_planszowej,
tu_wstaw_id_klienta,
'tu_wstaw_date_realizacji',
tu_wstaw_id_adresu,
tu_wstaw_liczbe_egzemplazy_danej_gry);
```

Sprawdzenie ID elementów o danej nazwie było już omawiane przy skryptach gry_planszowe, ogólny schemat wygląda następująco:

```
select id, nazwa_kolumny from nazwa_tabeli
where nazwa_kolumny = 'nazwa_elementu'
```

Usuwanie i modyfikowanie wierszy z tej tabeli jest zabronione regularnym pracownikom.

4.8 Wypozyczalnia

W celu dodania nowego wiersza należy wykonać następujące polecenie:

```
insert into wypozyczalnia(gra_planszowa,klient,data_wypozyczenia)
values(
id_gry_planszowej,
id_klienta,
'data wypozyczenia');
```

W celu modyfikacji daty oddania (która nie jest podawana przy akcie wypożyczenia) wypożyczenia o znanym ID należy wykonać polecenie:

```
update wypozyczalnia
set data_zwrotu = 'tu_wstaw_date_zwrotu'
where id = ID;
```

Podobnie jak tabela zakup usuwanie wierszy z tej tabeli jest zabronione regularnym pracownikom.

5 Widoki bazy danych

W celu łatwiejszej nawigacji przygotowano szereg widoków ułatwiający poruszanie się po bazie danych.

5.1 Dostępne gry

Widok ten pokazuje gry które są dostępne na stanie (mają więcej niż 0 dostępnych egzemplarzy).

```
create view dostepne_gry as
    select nazwa_gra as Dostepne_gry, liczba from gra_planszowa
    where liczba>0;
```

Kolejny widok pokazuje dostępne gry do wypożyczenia:

```
create view dostepne_gry_wypozyczenie as
    select nazwa_gra as Dostepne_gry from gra_planszowa
    where do_wypozyczenia is true
    and liczba>0;
```

I gry dostępne tylko na sprzedaż:

```
  create view dostepne_gry_sprzedaz as
    select nazwa_gra as Dostepne_gry from gra_planszowa
    where do_wypozyczenia is false
    and liczba>0;
```

Na koniec widok pokazujący niedostępne gry:

```
create view niedostepne_gry as
    select nazwa_gra as Niedostepne_gry from gra_planszowa
    where liczba = 0 or liczba = NULL;
```

5.2 Historia wypożyczeń

Kolejny widok pokazuje wszystkie gry które zostały wypożyczone wraz z klientami który je wypożyczyli formie przyjaznej dla obsługującego bazę danych (bez id):

```
create view historia_wypozyczen as
    select
    k.imie1||' '||k.nazwisko klient,
    w.data_wypozyczenia,
    w.data_zwrotu,
    g.nazwa_gra gra_planszowa
from
    wypozyczalnia w join gra_planszowa g
    on w.gra_planszowa=g.id
    join klient k
    on w.klient=k.id;
```

Kolejny widok pokazuje gry które zostały wypożyczone i nie zostały jeszcze oddane:

```
create view wypozyczone_gry as
    select
    k.imie1||' '||k.nazwisko klient,
    w.data_wypozyczenia,
    w.data_zwrotu,
    g.nazwa_gra gra_planszowa
from
    wypozyczalnia w join gra_planszowa g
    on w.gra_planszowa=g.id
    join klient k
    on w.klient=k.id
    where data_zwrotu is null;
```

6 Usunięcie bazy danych

Skrypt usuwający całą bazę danych:

```
drop view wypozyczone_gry;
drop view historia_wypozyczen;
drop view niedostepne_gry;
drop view dostepne_gry_wypozyczenie;
drop view dostepne_gry_sprzedaz;
drop view dostepne_gry;
drop table wypozyczalnia;
drop table zakup;
drop table klient;
drop table gra_planszowa;
drop table gatunek;
drop table autor;
drop table wersja_jezykowa;
drop table adres_wysylki;
```

7 Źródła

W trakcie wykonywania projektu korzystano z:

- materiałów z wykładów i laboratoriów udostępnionych w ramach przedmiotu Bazy Danych realizowanego w Europejskiej Uczelni w semestrze letnim 2024
- dokumentacji PostgreSQL znajdującej się na stronie https://www.postgresql.org/docs/current/
- narzędzia do generowania diagramów https://app.diagrams.net/

8 Załączniki

W załączniku znajdują się pliki *.sql zawierające skrypty tworzące bazę danych, wypełniające bazę danych przykładowymi danymi, tworzące i wywołujące poszczególne widoki, usuwające bazę danych.