Dokumentacja projektu bazodanowego

Przedmiot: Tworzenie aplikacji bazodanowych
Temat: Przychodnia

Autorzy: Jan Kucharski, Kamil Jarmoc

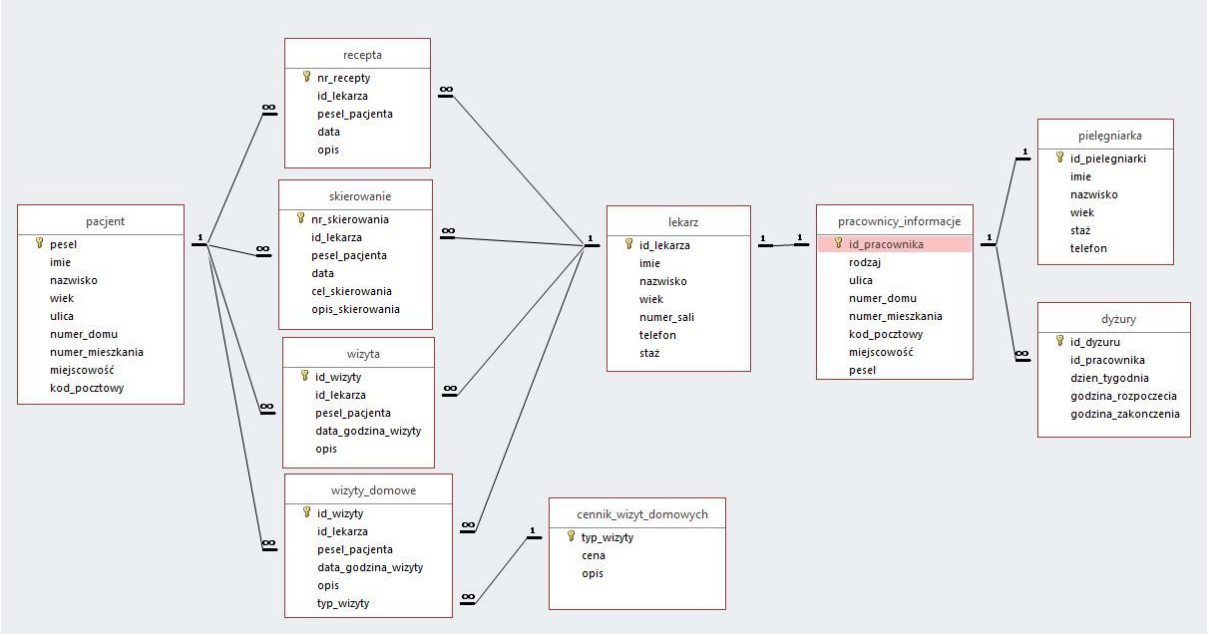
Prowadzący: dr inż. Małgorzata Krętowska

Wstęp

Celem projektu było stworzenie bazy danych dla przychodni. Jak wiadomo, w dobie komputeryzacji i automatyzacji, baza danych wraz z szybkim przeszukiwaniem jej, rejestrowaniem pracowników, pacjentów jest wręcz nieodzownym elementem. Wszystko musi być zapisane, by nie było różnych nieporozumień oraz w jasny i przystępny sposób przedstawiony potencjalnemu użytkownikowi systemu. Baza będzie przechowywać wszystkie potrzebne informacje o pacjentach, lekarzach, pielęgniarkach w celu właściwego funkcjonowania placówki medycznej. W skład wchodzą między innymi informacje o receptach, skierowaniach, czy też wizytach zarówno w placówce, jak i w domach pacjentów.

Projekt i implementacja aplikacji

Całościowy schemat z drobnymi modyfikacjami przedstawia poniższa grafika



Rys. Schemat bazy danych.

Charakterystyka tabel bazy danych

- **pacjenci**

Tabela pacjenci zawiera wszystkie informacje dotyczące osób korzystających z przychodni medycznej. Kluczem głównym dla tej relacji jest unikalny numer pesel, który jednoznacznie identyfikuje każdego pacjenta (każdego obywatela RP).

Atrybuty: pesel(liczba), **imię**(tekst), **nazwisko**(tekst), **wiek**(liczba), **ulica**(tekst), **numer_domu**(liczba), **numer_mieszkania**(numer), **miejscowość**(tekst), **kod_pocztowy**(liczba, która jest odpowiednio wprowadzana przy pomocy maski wprowadzania).

- **lekarz**

Tabela lekarze przechowuje dane personalne osób, które służą medyczną pomocą dla pacjentów. Zawiera ona podstawowe informacje, które są potrzebne w pierwszej kolejności. Więcej danych (dane adresowe i “poboczne”) są przedstawione w tabeli ‘pracownicy_informacje’. Z atrybutów godnych przedstawienia są **staż** (liczba lat przepracowanych w zawodzie) oraz **numer_sali** (sala, która jest przypisana dla każdego lekarza)
Kluczem głównym w tej tabeli jest **id_lekarza** - unikalny numer nadawany oddzielnie dla każdego lekarza.

Atrubuty: **id_lekarza**(liczba), **imie**(tekst), **nazwisko**(tekst), **wiek**(liczba), **numer_sali**(liczba), **telefon**(liczba), **staż**(liczba).

- **pielęgniarki**

Relacja ta przedstawia ważniejsze informacje nt. danych personalnych pielęgniarek. Kluczem głównym w tym przypadku będzie atrybut id_pielęgniarki - unikalny numer nadawany dla każdej pielęgniarki w przychodni.

Atrybuty: id_pielegniarki(liczba), **imię**(tekst), **nazwisko**(tekst), **wiek**(liczba), **staż**(liczba), **telefon**(liczba).

- **pracownicy_informacje**

Jak już wyżej zostało wspomniane, tabela ta przedstawia dodatkowe informacje na temat personelu medycznego. Chodzi o nic innego jak o dane adresowe, które z racji mniejszego znaczenia dla przychodni, zostały umieszczone w oddzielnej relacji.
Kluczem głównym jest id_pracownika (ogólne, które mieści w sobie id_pielęgniarki i id_lekarza). Warto zauważyć, że jest to tabela “przejściowa”, która łączy ze sobą tabele lekarz, pielęgniarka i dyżury.

Atrybuty: id_pracownika(liczba), **rodzaj**(wartość 0 to pielęgniarka, 1- lekarz). Przydatny atrybut przy identyfikacji danego pracownika. **ulica**(tekst), **numer_domu**(liczba), **numer_mieszkania**(liczba), **kod_pocztowy**(liczba), **miejscowość**(tekst), **pesel**(wartość tekstowa, która jest odpowiednio konwertowana - 11 liczb)

- **dyżury**

Zawiera informacje dotyczące dni pracy poszczególnych pracowników(pielęgniarek i lekarzy). Kluczem głównym jest identyfikator dyżuru. Kolejne atrybuty oznaczają jaki lekarz, w jakim dniu i w jakim wymiarze godzin pracuje.

Atrybuty: id_dużuru(liczba)**, id_pracownika**(liczba), **dzień_tygodnia**(tekst), **godzina_rozpoczecia**(tekst), **godzina_zakończenia**(tekst).

- **skierowanie**

Przechowuje dane o wypisanych skierowaniach przez konkretnego lekarza dla danego pacjenta. Kluczem głównym jest tutaj nr_skierowania, który jest oczywiście unikalny. Występują tutaj również klucze obce: id_lekarza i pesel_pacjenta. Pierwszy odnosi się do klucza głównego tabeli lekarz, a kolejny - tabela pacjent. Kolumny data i cel_skierowania są również obowiązkowe, aby skierowanie miało sens. Atrybut opis_skierowania może być opcjonalny (choć jest wskazany)

Atrybuty: nr_skierowania(liczba), **id_lekarza**(liczba), **pesel_pacjenta**(liczba), **data**(data), **cel_skierowania**(tekst), **opis_skierowania**(tekst).

- **recepta**

W tej tabeli są przechowywane informacje na temat recept przepisanych przez lekarzy. Kluczem głównym jest unikalny nr_recepty. Podobnie jak wyżej, klucze obce odwołują się do tych samych tabel (lekarz i pacjent).

Atrybuty: nr_recepty(liczba), **id_lekarza**(liczba), **pesel_pacjenta**(liczba), **data**(data), **opis**(tekst).

- **wizyta**

Relacja ta zawiera dane dotyczące umówionych wizyt pacjentów z poszczególnymi lekarzami. Kluczem głównym w tym przypadku jest id_wizyty, natomiast kluczami obcymi: id_lekarza oraz pesel_pacjenta. Atrybut data_godzina_wizyty służy do określenia w czasie wizyty, natomiast atrybut opis jest ewentualną, dodatkową informacją.

Atrybuty: id_wizyty(liczba), **id_lekarza**(liczba), **pesel_pacjenta**(liczba), **data_godzina_wizyty**(timestamp), **opis**(tekst).

- **wizyty_domowe**

Tak jak wyżej, z tą różnicą, że wizyty odbywają się u chorego, a nie w przychodni. Kluczem głównym jest id_wizyty, a kluczami obcymi są: id_lekarza, pesel_pacjenta oraz typ_wizyty.

Atrybuty: id_wizyty(liczba), **id_lekarza**(liczba), **pesel_pacjenta**(liczba), **data_godzina_wizyty**(timestamp), **opis**(tekst), **typ_wizyty**(tekst).

- **cennik_wizyt_domowych**

Zawiera informacje o kosztach takiej usługi oraz typie wizyty, który jest jednocześnie kluczem głównym.

Atrybuty: typ_wizyty(liczba), **cena**(liczba), **opis**(tekst)

Kody podprogramów

Kody podprogramów, które służą utworzeniu prawidłowej bazy danych znajdują się w pliku, który zostanie dołączony do dokumentacji. Znajdują się tam zarówno wyzwalacze, jak i skrypty tworzące i tabele, oraz wypełniające je przykładowymi danymi. Oczywiście też nie można było zapomnieć o sekwencjach, które służą do generowania kluczy głównych w większości tabel. Zaimplementowano także kilka innych podprogramów, które mają na celu zarówno prawidłowe funkcjonowanie bazy jak i sprawniejsze przeglądanie i modyfikacja.

Procedura na przenoszenie wizyt

```
create or replace procedure zmien_wizyte(id_lek number,
                                stara_data varchar2, data varchar2 )
is
data1 timestamp;
data2 timestamp;
begin
        data1:=to_timestamp (stara_data,'DD-MM-YYYY HH24:MI');
        data2:=to_timestamp (data,'DD-MM-YYYY HH24:MI');
        for i in (select * from wizyta ) loop
                update wizyta set data_godzina_wizyty= data2
                where i.id_lekarza=id_lek and data_godzina_wizyty= data1;
        end loop;
        for i in (select * from wizyta ) loop
                update wizyty_domowe set data_godzina_wizyty= data2
                where i.id_lekarza=id_lek and data_godzina_wizyty= data1;
        end loop;
end;
```

Celem powyższej procedury jest przenoszenie wizyt. Jak wiadomo, dany lekarz może być chory, przez co musi odwołać wizyty. Powyższa procedura umożliwia przeniesienie wszystkich wizyt z danego dnia i danego lekarza na inny dzień, w którym będzie mógł przyjąć danych pacjentów. Procedura dotyczy zarówno wizyt w przychodni jak i tych w domach pacjentów. Atrybutami tej procedury są:

- id_lek, czyli identyfikator chorego lekarza (lub takiego, który musi po prostu przełożyć wizytę),
- stara_data - planowana data wizyty, w którym terminie miała się odbyć, ale się nie odbędzie,
- data - nowa data, na którą zostaje dana wizyta przeniesiona.

Podprogram korzysta z 2 pętli kursorowych, które wyszukują wszystkie wystąpienia i odpowiednio modyfikują tabele wizyta oraz wizyty_domowe. Z racji tego, że przechowywane są też i godziny wizyt, wykorzystany jest typ timestamp, to należy przekazane daty sprowadzić właśnie do tego formatu zapisu.

Wyzwalacz w celu rozróżnienia dodawania lekarzy i pielęgniarek

```
create or replace trigger "PRACOWNICY_INFORMACJE_T1"
AFTER insert on "PRACOWNICY_INFORMACJE"
for each row
begin
	if :new.rodzaj = 0 then
		insert into pielegniarka(id_pielegniarki) values (:new.id_pracownika);
	end if;
	if :new.rodzaj = 1 then
		insert into lekarz(id_lekarza) values (:new.id_pracownika);
	end if;
end;
```

Wyzwalacz, który jest zamieszczony w pliku, który jest nałożony na tabelę ‘pracownicy_informacje’ tylko częściowo spełniał swoją funkcję. Dla 10 danych, które mieliśmy dodać wiedzieliśmy, kto był lekarzem, a kto pielęgniarką. Jednak takie rozwiązanie na dłuższą metę jest bardzo niepraktyczne. W tym celu stworzony został dodatkowy wyzwalacz, który miał za zadanie dopilnować, by odpowiednie id lekarza lub pielęgniarki trafiło do odpowiedniej tabeli. W tym celu dodatkowo została stworzona kolumna ‘rodzaj’, która jasno mówiła nam, czy dodajemy lekarza, czy pielęgniarkę. Po tym utworzyliśmy wyzwalacz, którego kod widoczny jest powyżej. Jak widać, został on nałożony jako AFTER, czyli po instrukcji. Po dodaniu rekordu do tabeli (wszak jeśli chcemy kogoś dodać, to od tej tabeli musimy zacząć) znana jest wartość pola ‘rodzaj’. Gdy zostanie to pole wypełnione to odpowiednie id ląduje w tabeli albo lekarza, albo pielęgniarki (korzystając z dodanego przed chwilą identyfikatora w tabeli ‘pracownicy_informacje’). Dzięki temu wiadomo, kto jest kim. Pozostaje jedynie wypełnić dalsze dane, gdyż wyzwalacz ten dodaje jedynie identyfikator. Resztę pól należy już dodać ręcznie.

Funkcja, która zwraca całkowity koszt pokryty przez pacjenta

```
create or replace function suma(ps varchar)
return float
is
summ float;
begin
	select sum(c.cena) into summ from pacjent p, wizyty_domowe w,
	cennik_wizyt_domowych c
	where p.pesel=ps and w.pesel_pacjenta=p.pesel and
	c.typ_wizyty=w.typ_wizyty;
return summ;
end;
```

Powyższa funkcja służy do zwrócenia całkowitego kosztu wizyt domowych zadanego pacjenta (tylko za wizyty domowe, czyli prywatne, się płaci). Funkcja wylicza sumę na podstawie jasno sprecyzowanego cennika wizyt domowych/

Ciekawe rozwiązania implementacyjne

Z ciekawych rozwiązań implementacyjnych to należy uwzględnić ten dodatkowy wyzwalacz, który automatyzuje proces dodawania danego pracownika.
W aplikacji występują też LOVy, które bardzo upraszczają i dają jasny przekaz tego, kogo lub co dodajemy. Ponadto znajdują się też pola wyboru (właśnie przy wyborze, czy jest to lekarz, czy pielęgniarka) oraz pola, w których można wybrać datę. Ciekawym rozwiązaniem są raporty (widoki), które pod spodem po wybraniu danego pacjenta ukazują wyniki wyszukiwania. Wszystkie te funkcjonalności zostaną przedstawione przy omawianiu interfejsu użytkownika.

Przedstawienie aplikacji

Interfejs aplikacji miał za zadanie uprościć podstawowe operacje na bazie danych. Został on tak stworzony, by każdy, kto będzie miał dostęp, mógł w swobodny i łatwy sposób wyszukiwać, wstawiać i modyfikować informacje. Interfejs został utworzony w środowisku Oracle Apex.

Praktycznie całość aplikacji została wyposażona w interaktywne formularze łącznie z raportami, które w niebagatelny sposób upraszczają stosowanie i wszelkie operacje na danych.

Strona główna prezentuje się następująco.

Rys. Strona główna aplikacji.

Strona główna w bazie pełni rolę po prostu czysto informacyjną. Jak widać, po lewej stronie znajduje się interaktywny pasek, z poziomu którego możemy przechodzić do różnych tabel i funkcjonalności. Klikając w niego od razu przenosimy się do interesującej nas relacji.

Rys. Rozwijana lista w menu.

Niektóre pozycje zawierają rozwijaną listę na pasku menu. W ten sposób całość zachowuje ład i porządek. W ramach odnośnika “Zarządzanie bazą danych” możemy przenieść się do odnośnika “Informacje o pracownikach”, w ramach którego występują tabele: ’pracownicy_informacje’, ‘lekarz’, ‘pielęgniarka’ i ‘dyżury’. Na uwagę zasługuje pasek, w którym możemy wpisać interesującą nas frazę i automatycznie wyszuka rekordy zawierające dany zasób.

Rys. Opcje ‘Actions’.

Rozwijane pole ‘Actions’ daje ogromną ilość narzędzi do przeszukiwania danych. Można między innymi filtrować, grupować dane, sortować i wiele innych.

Po kliknięciu w ikonkę ołówka zostaniemy przeniesieni do edycji danego rekordu.

Rys. Okno edycji pracowników.

W ten oto sposób możemy modyfikować dany rekord. Na uwagę zasługują 2 pola radio, z których można wybrać tylko jedno. Odpowiada to polu rodzaj (czyli 0 lub 1). Jak widać, taki zapis bardzo ułatwia pracę przy modyfikacji rekordów. Aby zapisać dane należy kliknąć przycisk “Zapisz zmiany”. W innym przypadku możemy anulować zmiany, poprzez wciśnięcie “Anuluj”, które znajdują się na tej samej stronie. Możemy dodatkowo usunąć danego pracownika poprzez wciśnięcie “Usuń pracownika”. Cofając się do “Pracownicy informacje” i kliknięcie w przycisk “Dodaj pracownika” przeniesie nas do ekranu, który znajduje się poniżej:

Rys. Formularz tworzenia pracownika.

W tym formularzu podajemy dane, po czym możemy utworzyć pracownika, lub anulować tworzenie.

Ciekawym udogodnieniem dla pracowników jest rozwijana lista korzystająca z LOV. Aby skorzystać z niej, należy wejść m. in. w zakładkę Wizyty i rozpocząć edycję lub dodawanie rekordu za pomocą albo przycisku “Dodaj wizytę” albo symbolu ołówka.

Rys. Okno edycji wizyty.

Po kliknięciu w rozwijaną listę w polu lekarz lub pacjent wyświetli nam się lista osób (imiona i nazwiska). Dzięki temu nie musimy znać identyfikatorów czy też peseli pacjentów. Jest to znaczne ułatwienie i przyśpieszenie pracy. Możemy zauważyć też, że w polu ‘Data Godzina WIzyty’ mamy możliwość wybrania daty. Nie musimy się przejmować tym, że źle wpiszemy - można w bardzo łatwy sposób ewentualnie poprawić.

Po wejściu w “Zarządzanie bazą danych” -> “Wszystkie wizyty” -> “Zmień daty wizyt lekarza” przeniesieni zostaniemy do następującego okna

Rys. Okno zmiany wizyty

Mamy możliwość zmiany wizyty danego lekarza (przeniesienie jego wizyt). Należy wybrać z rozwijanej listy lekarza, podać datę, na którą zostaną przeniesione wizyty, oraz datę, z której zostaną przeniesione na nowy termin.

Kolejną ciekawą zaimplementowaną funkcją, są raporty widokowe. Aby zobaczyć, jak wyglądają, należy w głównym pasku rozwinąć “Raporty o pacjentach”. Przykładowy widok prezentuje się następująco.

Rys. Okno widoku.

Dla każdej zakładki widok wyszukuje zadaną informację o pacjencie. W tym przypadku po znalezieniu danego pacjenta i kliknięciu w jego na górze, na dole wyświetli się raport z wszystkimi wizytami tego pacjenta (które były, jak i te, które się odbędą). W znaczny sposób upraszcza to wyszukiwanie zadanych danych.

W tej samej zakładce (“Raporty o pacjentach”) możemy wejść w zakładkę “Informacje o kosztach”. Widok przedstawi się następująco:

Rys. Funkcja na całkowity koszt wizyt pacjenta.

Po wybraniu z listy LOV interesującego nas pacjenta i kliknięciu w przycisk “Zatwierdź” jako wynik otrzymamy całkowity koszt wizyt domowych przez danego pacjenta.

Podsumowanie

W ramach aplikacji został utworzony przejrzysty interfejs użytkownika. Żaden pracownik nie będzie miał problemu z jego obsługą. Sekcja “Zarządzanie bazą danych” Odpowiada za modyfikację, usuwanie, wstawianie nowych danych, zaś sekcja “Raporty o pacjentach” ma za zadanie w łatwy sposób wyświetlenie wszystkich ważnych informacji o pobycie i “życiu” pacjentów przychodni. Za pomocą rozwijanych list można bardzo łatwo przeprowadzać operacje, a dodatkowe podprogramy dopełniają całość.