Autorzy:
Bartosz Cybulski, Bartłomiej Królikowski

Nazwa projektu:
Aplikacja bazodanowa

Cel aplikacji:

analiza informacji o strukturze (pracownicy, zespoły, działy), działalności (projekty), finansach (zamówienia), klientach i dostawcach firmy oraz wpływanie na jej organizację

Wymagania:

- możliwość zobaczenia informacji osobistych przez pracownika
- członkowie zespołów mogą zobaczyć podstawowe informacje o realizowanym projekcie
- zarządcy zespołów mogą zobaczyć wszystkie informacje o realizowanym projekcie i członkach zespołu oraz ustawiać wynagrodzenia i czas pracy pracowników oraz dodawać, usuwać i przenosić członków zespołu
- zarządcy działów mogą oglądać wszystkie informacje na temat swojego działu, przyjmować zlecenia od klientów, dobierać dostawców i produkty, delegować zespoły do projektów, wyznaczać budżety i tygodniowe czasy pracy projektów, decydować o statusie projektu oraz ustalać zarządców zespołów
- szef firmy może zobaczyć wszystkie informacje o firmie, modyfikować dane i historię pracowników, zwalniać i przyjmować pracowników, ustalać zarządców działów, ustalać, które działy realizują zlecenie, zmieniać strukturę firmy (dodawanie / usuwanie działów)

Ograniczenia:

- pracownicy mogą oglądać informacje tylko o swoich podwładnych
- całkowity tygodniowy czas pracy zespołu jest sumą czasów pracy poszczególnych pracowników
- wynagrodzenia członków projektu są częścią budżetu projektu, więc budżet nie może być mniejszy od sumy wynagrodzeń członków

Funkcjonalności:

- aplikacja pozwala na połączenie się pracownika z systemem w celu wykonania dozwolonych działań
- aplikacja przy tworzeniu projektu automatycznie sprawdza czy przydzielony budżet jest wystarczająco duży by wypłacić z niego pensje wszystkim pracownikom biorącym udział w projekcie, jeśli nie to nie pozwala na utworzenie projektu
- aplikacja przy zmianie pensji pracownika automatycznie sprawdza czy budżet nie jest przekroczony, jeśli tak to zmiana jest cofana

- aplikacja pozwala na ocenę opłacalności projektu (na podstawie różnicy między budżetem a dochodem od klientów)
- aplikacja pozwala na tworzenie kopii zapasowych bazy i odtwarzanie zapisanych baz

Opis encji :

1. **Działy :**
   Krotki w tej encji reprezentują poszczególne działy w firmie. Każda krotka posiada 3 atrybuty :id_działu , zarządca_działu,nazwa_działu. Np : (1,15490,Dział Telekomunikacji) => Dział o nazwie "Dział Telekomunikacji" indeksie 1 zarządzany przez pracownika o indeksie 15490 .
   Atrybuty :
   - id_działu : jest to nieujemny Integer oraz klucz główny encji. Każdy dział posiada swój własny unikalny numer identyfikacyjny , który zawiera się w atrybucie id_działu.
   - zarządca_działu : jest to nieujemny Integer oraz klucz obcy encji , przechowuje numer identyfikacyjny pracownika , który sprawuje funkcje zarządcy działu.
   - nazwa_działu : jest to varchar(90) , przechowywana jest w nim nazwa działu.
2. **Zespoły:**
   Krotki w tej encji reprezentują poszczególne zespoły znadujące się w firmie. Zespół to pewna grupa pracowników , którzy współpracują razem nad projektem bądź projektami. Każdy dział posiada swoje własne zespoły. Każda krotka posiada 6 atrybutów :
   id_zespołu ,zarządca_zespołu,nazwa_zespołu,dział,budżet,godziny_tygodniowo. Np. (184,22,Alfa,1,20000,120) => Zespół o indeksie 184 i nazwie "Alfa", zarządzany przez pracownika o indeksie 22 , należy do działu o indeksie 1, posiada budżet 20000, wszyscy członkowie zespołu pracują łącznie 120 godzin nad realizujacą projektów/projektu tego zespołu.
   Atrybuty :
   - id_zespołu : jest to nieujemny Integer oraz klucz główny encji. Każdy zespół posiada swój własny unikalny numer identyfikacyjny , który zawiera się w atrybucie id_zespołu.
   - zarządca_zespołu : jest to nieujemny Integer oraz klucz obcy encji. Każdy zespół posiada jednego zarządce , który nadzoruje prace w zespole , klucz identyfikacyjny pracownika , który pełni tę funkcję znajduje się w atrybucie zarządca_zespołu.
   - nazwa_zespołu : jest to varchar(90), przechowywana jest w nim nazwa zespołu.
   - dział : każdy zespół podlega pod pewien dział (jeden dział może mieć wiele zespołów natomiast każdy zespół należy tylko do jednego działu). W atrybucie dział przechowywany jest numer identyfikacyjny działu do , którego należy dany zespół.
   - budżet : jest to nieujemny Integer w , którym przechowywany jest budżet dla całego zespołu. Budżet jest sumą budżetów wszystkich projektów , których podejmują się zespoły oraz wypłacane jest z niego wynagrodzenie dla każdego członka danego zespołu.
   - godziny_tygodniowo : jest to nieujemny Integer w , którym przechowywana jest informacja o sumie godzin pracy nad projektami zespołu każdego członka zespołu. Zarządca zespołu (bądź jego przełożony)ustala wartość atrybutu godziny_tygodniowo a następnie rozdziela daną liczbę godzin każdemu członkowi zespołu. Procedura rozdzielająca powinna działać w ten sposób , że nie jest możliwe zatwierdzenie godzin pracy dla wszystkich członków

jeżeli suma rozdanych godzin != wartość atrybutu godziny_tygodniowo dla danego zespołu.

3. **Grupy_zespołu :**
   Jest to specjalna encja bez klucza głównego , która wiąże ze sobą encje Zespoły oraz Pracownik. Przechowuje informacje , którzy pracownicy należą do poszczególnych zespołów ponadto przewiduje możliwość, w której jeden pracownik (bardzo nadgorliwy) należy do kilku zespołów. Ponadto zawierane są w niej także informacje o byłych członkach zespołów dzięki czemu pracownik posiada możliwość przejrzenia swojej historii w firmie. Każda krotka posiada atrybuty : zespol , pracownik , status,wynagrodzenie,godziny. Np : (22,1238,Obecny,1000,40) => w zespole o indeksie 22 znajduje się (posiada status "Obecny") pracownik o indeksie 1238 , za swoją pracę w zespole pobiera tygodniowo 1000 zł wynagrodzenia (z budżetu zespołu)oraz pracuje 40 godzin tygodniowo nad projektami zespołu.
   Atrybuty :
   - zespol : jest to nieujemny Integer oraz klucz obcy. W tym atrybucie przechowywane są informacje o numerach identyfikacyjnych zespołów.
   - pracownik : jest to nieujemny Integer oraz klucz obcy. W tym atrybucie przechowywane są informacje o numerach identyfikacyjnych pracowników.
   - status : jest to varchar(90). W tym atrybucie przechowywana jest informacja o statusie pracownika w danym zespole np. : "Obecny" , "Usunięty", "Przeniesiony", "Zarządca" . Podczas usuwania pracownika z zespołu wartośc atrybutu w krotce odpowiadającej za przynależność pracownika zostaje zmieniona na "Usunięty".
   - wynagrodzenie : jest to nieujemny Integer. W tym atrybucie przechowywana jest informacja o pobieranym wynagrodzeniu z budżetu z pewnego zespołu (np. zespół X) przez pewnego pracownika (np. pracownik Y). Wartość tego atrybutu ustala zarządca zespołu X. W chwili usunięcia pracownika z zespołu wartość jest zmieniana na 0.
   - godziny : jest to nieujemny Integer. W tym atrybucie przechowywana jest informacja o tygodniowej liczbie godzin jaka pracownik powinien poświęcić na realizacje projektów zespołu. Wartośc tego atrybutu ustala zarządca zespołu. W chwili usunięcia pracownika z zespołu wartość jest zmieniana na 0.

4. **Pracownik :**
   Krotki w tej encji reprezentują poszczególnych pracowników zatrudnionych przez szefa firmy. Każda krotka posiada 5 atrybutów : id_pracownika , imie, nazwisko , wypłata , godziny_tygodniowo. Np. : (384,Jan,Kowalski,2000,40) => Pracownik o indeksie 384 , imieniu Jan, nazwisku Kowalski , który zarabia łącznie 2000 zł tygodniowo oraz pracuje 40h.
   Atrybuty :
   - id_pracownika : jest to nieujemny Integer oraz klucz główny encji. Każdy pracownik posiada swój własny unikalny numer identyfikacyjny , który jest przechowywany w tym atrybucie.
   - imie : jest to varchar(90) . W tym atrybucie przechowywane jest imie danego pracownika.

- nazwisko : jest to varchar(90). W tym atrybucie przechowywane jest nazwisko danego pracownika.
- wypłata : jest to nieujemny Integer. W tym atrybucie zawierana jest suma wynagrodzeń danego pracownika ze wszystkich zespołów do , których należy pracownik (każe wynagrodzenie z encji Grupy_zespołów dodawane jest do wypłate dopiero po zatwierdzeniu przez zarządce każdego zespołu do , których należy pracownik tzn. Jeżeli pracownik A należy do zespołów X i Y w , których pobiera odpowiednio 2000zł i 1000zł , po zatwierdzeniu wynagrodzeń przez zarządce zespołu X wartośc atrybutu wypłata wynosi 2000zł natomiast nastepnie po zatwierdzeniu wynagrodzeń przez zarządce zespołu Y wartośc atrybutu wypłata wynosi 3000zł).
- godziny_tygodniowo : jest to nieujemny Integer. W tym atrybucie zawierana jest suma godzin danego pracownika jaką musi odpracować tygodniowo we wszystkich swoich zespołach.

5. **Loginy :**
   W tej encji przechowywane są loginy wszystkich pracowników firmy , aby umożliwić im zalogowanie się do aplikacji. Z uwagi na bezpieczeństwo hasła przechowywane są w innej encji. Każda krotka posiada 2 atrybuty : id_pracownika, login . Np. (355, Skoczek) => pracownik o id 355 posiada login Skoczek.
   Atrybuty :
   - id_pracownika : jest to nieujemny Integer oraz klucz główny encji. W tym atrybucie przechowywane są numery identyfikacyjne wszystkich pracowników z encji Pracownik.
   - login : jest to varchar(90) . W tym atrybucie znajduje się login pracownika.
6. **Hasla :**
   W tej encji przechowywane są hasła wszystkich pracowników firmy , aby umożliwić im zalogowanie się do aplikacji. Z uwagi na bezpieczeństwo loginy przechowywane są w innej encji. Każda krotka posiada 2 atrybuty : id_pracownika , hash_hasla. Np. (355,71ca46a81543dbd0a1435b9683e44f68) => pracownik o id 355 posiada hash_hasla 71ca46a81543dbd0a1435b9683e44f68.
   Atrybuty :
   - id_pracownika : jest to nieujemny integer oraz klucz główny encji. W tym atrybucie przechowywane są numery identyfikacyjne wszystkich pracowników z encji Pracownik.
   - hash_hasla : jest to varchar(90). W tym atrybucie znajduje się hash hasła pracownika.
7. **Projekty :**
   W tej encji przechowywane są wszystkie projekty , których podejmuje bądź podjęła się firma. Każdy projekt dotyczy jednego zlecenia oraz realizowany jest przez jeden zespół. Jednak wiele projektów może dotyczyć tego samego zlecenia oraz jeden zespół może realizować wiele projektów. Każda krotka posiada 6 atrybutów : id_projektu ,status , zespol, zlecenie, przydzielony_budzet, nazwa_projektu. Np. (555,Realizowany,382,22,30000,Zaprojektowanie aplikacji bazodanowej) => Projekt o indeksie 555 jest realizowany przez zespol o indeksie 382 , odnosi się do zlecenia o indeksie 22 oraz przeznaczony na niego

tygodniowy budzet wynosi 30000 zł, projekt nazywa się ,,Zaprojektowanie aplikacji bazodanowej".
Atrybuty :

- id_projektu : jest to nieujemny integer oraz klucz główny encji. Każdy projekt posiada swój własny unikalny numer identyfikacyjny , który przechowywany jest w tym atrybucie.
- status : jest to varchar(90) określający status projektu np.: czy projekt został "Porzucony", "Zrealizowany" czy jest "Realizowany" , warto zauważyć , że ponieważ wiele projektów mopże dotyczyć jednego zlecenia może dojść do sytuacji w , której np.: zespoły A B C realizuja odpowiednio projest Q W E dotyczące zlecenia X , następnie po pewnym czasie zespół C jako pierwszy zrealizował swój projekt , status projektu E zmienia się na "Zrealizowany" natomiast projektów Q i W na "Porzucony"
- zespol : jest to nieujemny integer oraz klucz obcy encji, atrybut zawiera numer identyfikacyjny zespołu realizującego dany projekt.
- zlecenie : jest to nieujemny integer oraz klucz obcy encji , atrybut zawieta numer identyfikacyjny zlecenia , którego dotyczy projekt.
- przydzielony_budzet : jest to nieujemny integer, zawiera informacje o tygodniowym budżecie przydzielonym przez zarządce działu na realizację danego projektu. Suma przydzielonych budżetów do projektów realizujących pewne zlecenie nie powinna przekraczać wartości zlecenia.
- nazwa_projektu : jest to varchar(90) określający nazwę danego projektu.

8. **Zlecenia :**
   W tej encji przechowywane są wszystkie zlecenia , które przyjęła firma. Każde zlecenie przypisane jest do jednego działu oraz jednego klienta. Każda krotka posiada 4 atrybuty : id_zlecenia , klient , wartosc_zlecenia, dzial. Np. (484,32,40000,3) => Zlecenie nadaje przez klienta 32 o wartosci 40000 zł przypisane do działu o indeksie 3.
   Atrybuty :
   - id_zlecenia : jest to nieujemny integer oraz klucz główny encji. Każde zlecenie posiada swój własny unikalny numer identyfikacyjny , który przechowywany jest w tym atrybucie.
   - klient : jest to nieujemny integer oraz klucz obcy encji. Każde zlecenie przypisane jest do jednego klienta , którego numer identyfikacyjny zapisany jest w atrybucie klient.
   - wartość_zlecenia : jest to integer , okreslający na początku ile klient zapłacił za zlecenie następnie wartość tego atrybutu jest zmniejszana co tydzień o sumę budżetów wszystkich projektów dotyczących tego zlecenia.
   - dzial : jest to nieujemny integer oraz klucz obcy encji. Każde zlecenie przypisane jest do konkretnego działu , którego numer identyfikacyjny zapisane jest w atrybucie dział.
9. **Klienci :**
   W tej encji przechowywane są informacje o wszystkich klientach , którzy nawiazali współpracę z firmą. Każda krotka posiada 2 atrybuty : id_klienta , nazwa_klienta. Np. (33, Znana Firma Informatyczna) => firma nawiazała

współpracę z klientem nazywającym się "Znana Firma Informatyczna" i przydzieliła mu id_klienta = 33.
Atrybuty :

- id_klienta : jest to nieujemny integer oraz klucz główny encji. Każdy klient posiada swój własny unikalny numer identyfikacyjny który przechowywany jest w tym atrybucie.
- nazwa_klienta : jest to varchar(90). W tym atrybucie znajduje się nazwa klienta , może być to osoba prywatna "Nazwisko Imie" jak i firma "Nazwa Firmy".

10. Zakupy:

W tej encji przechowywane są wszystkie produkty kupowane aktualnie przez dział. Każdy zakup jest przeznaczony dla jednego działu i dostarczany przez jednego dostawcę. Każda krotka posiada 5 atrybutów : id_zakupu, dział, dostawca, produkt wartość_zakupu. Np. (123, 4, 56, „Nowy program”, 70000) => zakup do działu 4 od dostawcy 56 produktu „Nowy program” o wartości 70000zł
Atrybuty:

- id_zakupu : jest to nieujemny int oraz klucz główny encji.
- dział : jest to nieujemny int – klucz obcy wskazujący zamawiający dział
- dostawca : nieujemny int – klucz obcy wskazujący dostawcę produktu
- produkt : varchar – nazwa zamawianego produktu
- wartość_zakupu : kwota, jaką dział musiał zapłacić za produktu

11. Zakupy:

W tej encji przechowywane są wszyscy aktualni dostawcy działów. Każda krotka posiada 2 atrybuty : id_dostawcy, nazwa_dostawcy. Np. (12, „Firma elektroniczna”) => pewien dział kupuje produkty od dostawcy o nazwie „Firma elektroniczna”.
Atrybuty:

- id_dostawcy : nieujemny int – klucz główny
- nazwa_dostawcy : varchar – nazwa zapisanego dostawcy (firma, lub osoba prywatna)

Opis procedur i funkcji dostępnych dla poszczególnych typów użytkowników :

Zwykły pracownik :

Podczas wywoływania dostępnych procedur korzysta z paremetru @prac , który jest typu Integer oraz oznacza indeks pracownika. **ZWYKŁY PRACOWNIK NIE MOŻE GO ZMIENIAĆ !**

- Liczba_godzin_pracownika(@prac) : Funkcja zwraca łączną sumę godzin pracownika oraz korzystając z relacji wyświetla ile godzin tygodniowo pracownik pracuje dla poszczególnego zespołu.
- Wynagrodzenie_pracownika(@prac) : Funkcja zwraca łączną sumę wynagrodzenia pracownika oraz korzystając z relacji wyświetla ile pracownik zarabia podczas pracy w poszczególnym zespole.
- Zespoły_pracownika(@prac) : Procedura wyświetla wszystkie zespoły do, których należy oraz należał pracownik wraz ze swoim obecnym statusem w nich.
- Projekty_pracownika(@prac) : Procedura wyświetla wszystkie projekty , które realizuje oraz realizował pracownik wraz z obecnym statusem tych projektów.
- Dane_pracownika(@prac) : Procedura wyświetla dane pracownika , w przypadku błędu w danych należy skontaktować się z odpowiednią instancją.
- Historia_pracownika(@prac) : Procedura wyświetla wszystkie zespoły oraz realizowane przez nich projekty wraz ze statusem projektów. (Jest to przydatne w momencie , kiedy pracownik np. : ubiega się o dołączenie do nowego zespołu w firmie i wraz z podaniem wysyła wygenerowaną przez aplikacje historie).

Zarządca zespołu :

Należy pamiętać o tym , że zarządca zespołu także jest pracownikiem => może korzystać z funkcjonalności aplikacji jak zwykły pracownik. Natomiast z powodu posiadanych obowiązków dostaje także nowe możliwości. Zadaniem zarządcy zespołu jest nadzorowanie pracy określonego zespołu. Podczas wywoływania swoich procedur korzysta z parametrów : @prac - do funkcjonalności dla zwykłego pracownika , @zesp - jest to parametr typu Integer oraz oznacza indeks zespołu , @cz - jest to parametr typu Integer oraz oznacza indeks członka zespołu, @wyn - Nie ujemny integer określający wysokośc wynagrodzenia dla poszczególnego członka , @czas - Nie ujemny integer określający czas pracy. Zarządca może ustalić wartość parametru @cz , @wyn oraz @czas jednak **NIE MOŻE ZMIENIIAĆ PARAMETRÓW @prac i @zesp!**

- Budżet_zespołu(@zesp) : Funkcja zwraca łączny budżet zespołu , który jest sumą przydzielonych budżetów do wszystkich projektów realizowanych przez zespół.

- Członkowe_zespołu(@zesp) : Procedura korzystając z tabeli Grupy_zespołów wyświetla wszystkich pracowników , którzy znajdują się w danym zespole.
- Ustal_wynagrodzenie(@zesp,@cz,@wyn) : Procedura modyfikuje tabele Grupy_zespołów, do atrybutu "wynagrodzenie" wpisuje wartośc @wyn w krotce gdzie atrybut "zespół" = @zesp oraz atrybut "pracownik" = @prac
- Ustal_czas_pracy_członka(@zesp,@czas,@wyn) : Procedura modyfikuje tabele Grupy_zespołów, do atrybutu "godziny" wpisuje wartośc @czas w krotce gdzie atrybut "zespół" = @zesp oraz atrybut "pracownik" = @prac.
- Ustal_czas_pracy(@zesp,@czas) : Procedura modyfikuje tabele Zespoły , do atrybutu "godziny_tygodniowo" wpisuje wartośc @czas w krotce gdzie atrybut "zespół" = @zesp.
- Zatwierdź_wynagrodzenie(@zesp) : Funkcja działa jako transakcja do atrybutu "wypłata" w tabeli Pracownik dodaje wartość atrybutu "wynagrodzenie" z tabeli Grupy_zespołów , wszędzie tam gdzie Pracownik.id_pracownika = Grupy_zespołów.pracownik i Grupy_zespołów.zespol = @zesp. Jeżeli jednak suma wynagrodzeń przekracza budżet zespołu transakcja kończy się niepowodzeniem.
- Zatwierdź_czas_pracy(@zesp) : Funkcja działa jako transakcja do atrybutu "godziny_tygodniowo" w tabeli Pracownik dodaje wartość atrybutu "godziny" z tabeli Grupy_zespołów , wszędzie tam gdzie Pracownik.id_pracownika = Grupy_zespołów.pracownik i Grupy_zespołów.zespol = @zesp. Jeżeli jednak suma godzin przekracza budżet zaplanowana sumęgodzin pracy dla całego zespołu transakcja kończy się niepowodzeniem.
- Dodaj_do_zespołu(@zesp,@cz) : Procedura dodaje do zespołu pracownika o indeksie @cz. Atrybut "status" w tabeli Grupy_zespołów w krotce gdzie "pracownik" = @cz oraz "zespol" = @zesp zmienia się na "Obecny". (W przypadku dodania nie odpowiedniej osoby należy skontaktować się z odpowiednią instancją aby poprawić zmiany w historii).
- Usuń_z_zespołu(@zesp,@cz) : Procedura usuwa z zespołu pracownika o indeksie @cz. Atrybut "status" w tabeli Grupy_zespołów w krotce gdzie "pracownik" = @cz oraz "zespol" = @zesp zmienia się na "Usunięty". (W przypadku usunięcia nie odpowiedniej osoby należy skontaktować się z odpowiednią instancją aby poprawić zmiany w historii).
- Przenieś_z_zespołu(@zesp,@cz) : Procedura usuwa z zespołu pracownika o indeksie @cz. Atrybut "status" w tabeli Grupy_zespołów w krotce gdzie "pracownik" = @cz oraz "zespol" = @zesp zmienia się na "Przeniesiony". (W przypadku przeniesienia nie odpowiedniej osoby należy skontaktować się z odpowiednią instancją aby poprawić zmiany w historii).

- Projekty_zespołu(@zesp) : procedura wyświetla wszystkie projekty w , których bierze bądź brał udział zespół @zesp.

Zarządca działu :

Należy pamiętać o tym , że zarządca działu jest także pracownikiem => może korzystać z funkcjonalności aplikacji jak zwykły pracownik. Natomiast z powodu posiadanych obowiązków dostaje także nowe możliwości. Zadaniem zarządcy działu jest nadzorowanie pracy określonego działu poprzez zarządzanie zleceniami i zespołami. Ponadto zarządca działu może sprawdzać oraz modyfikować wewnętrzne ustawienia zespołu. Podczas wywoływania swoich procedur korzysta z parametrów : @prac - do funkcjonalności dla zwykłego pracownika , @zesp - jest to parametr typu Integer oraz oznacza indeks zespołu , @cz - jest to parametr typu Integer oraz oznacza indeks członka zespołu, @wyn - Nie ujemny integer określający wysokośc wynagrodzenia dla poszczególnego członka , @czas - Nie ujemny integer określający czas pracy , @kl - Nie ujemny integer określający id klienta, @dost – varchar określający nazwę dostawcy, @nazwa - varchar(90) okreslajacy nazwe , @wart - Integer określający wartość , @zl - nie ujemny integer określający id zlecenia , @pr - nie ujemny integer określający id projektu , @dz - nie ujemny integer określający id działu, @st - varchar(90) określający status projektu. Zarządca może ustalić wartość parametru @zesp, @cz , @wyn , @czas , @kl, @dost ,@nazwa, @wart, @zl ,@pr , @st jednak **NIE MOŻE ZMIENIIAĆ PARAMETRÓW @prac i @dz!**

- Dodaj_klienta(@nazwa) : Procedura dodaje do tabeli Klienci nowy wiersz gdzie Klienci.nazwa_klienta = @kl_nazwa.
- Dodaj_zlecenie(@kl,@wart,@dz) : Procedura dodaje do tabeli Zlecenia nowy wiersz gdzie Zlecenia.klient = @kl , Zlecenia.dzial = @dz oraz Zlecenia.wartosc_zlecenia = @wart_zl.
- Zakup_produkt(@nazwa, @dost) : Procedura dodaje produkt o nazwie @nazwa, i łączy go z dostawcą o nazwie @dost (dodając takiego jeśli nie istnieje w tabeli)
- Utworz_projekt(@nazwa,@zl,@zesp) : Procedura dodaje do tabeli Projekty nowy wiersz gdzie Projekty.zlecenie = @zl oraz Projekty.nazwa_projektu = @nazwa oraz Projekty.zespol = @zesp. Procedura nie wykonuje się jeżeli zespol o indeksie @zesp nie należy do działu.
- Ustal_budżet_projektu(@pr , @wart,@dz) : Procedura modyfikuje tabele Projekty , w miejscu gdzie "id_projektu" = @pr w atrybucie "przydzielony_budzet" wpisuje wartosc @wart.Procedura nie wykonuje się jeżeli zlecenie przydzielone do projektu (Projekt.zlecenie nie należy do działu => porownuje odpowiadajacy atrybut Zlecenia.dzial z @dz w miejscu gdzie

Zlecenia.id_zlecenia = Projekty.zlecenie i Projekty.id_projektu = @pr)

- Sprawdź_bilans(@dz) : Funkcja wyświetla wszystkie zlecenia podjete przez dział oraz ich wartosc, wszystkie projekty realizowane przez dzial oraz ich budżety wraz z zespołami które je realizuja oraz wszystkie zespoły znajdujące się w dziale oraz ich budżety. Nastepnie od sumy wszystkich zleceń odejmuje sume budzetow wszystkich zespołów i pokazuje wynik.
- Zlecenenie_info(@zl ,@dz) : Procedura jeżeli Zlecenia.dział = @dz wyswietla z tabeli Zlecenia w miesjcu gdzie "id_zlecenia" = @zl atrybut "wartosc_zlecenia" oraz "klient" oraz z tabeli Projekty w miejscu gdzie Projekty.zlecenie = @zl wszystkie informacje z tabeli.
- Zmień_status(@pr , @st, @dz) : Procedura modyfikuje tabele Projekty w miejscu gdzie atrybut "id_projektu" = @pr ustawia atrybut "status" na wartosc @st. Jeżeli projekt jest wykonywany przez zespół , którego dział = @dz.
- Zatwierdź_budżet(@dz) : Funkcja działa jako transakcja , modyfikuje wartosci tabeli Zespoly.budzet ustawiajac sume z wartosci z Projekty.przydzielony_budzet wszedzie tam gdzie Zespoly.id_zespolu = Projekty.zespol.
- ZM_zarzadcy_zespolu(@zesp,@cz,@dz) : Procedura zmienia status obecnego zarządcy zespołu o indekscie @zesp na "Były zarządca" oraz ustanawia członka zespołu o indeksie @cz nowym zarządcą zmieniając mu asttaus na "Zarządca". Procedura nie wykonuje się jeżeli dział zespołu != @dz.
- Utworz_zespol(@cz,@nazwa,@dz) : Procedura dodaje nowi wiersz w tabeli Zespoły , do atrybutu "zarządca_zespołu" przypisuje wartość @cz, do atrybutu "nazwa_zespołu" przypisuje wartość @nazwa, do atrybutu "dział" przypisuje wartość @dz.
- Rozwiaz_zespol(@zesp,@dz) : Procedura jeżeli zespół o indeksie @zesp należy do działu @dz zmienia status wszystkich projektów zespołu o indeksie @zesp na "Porzucony" oraz przenosi wszystkich członków zespołu @zesp.

Szef :

Jest to osoba z najwyższym poziomem dostepu równym administratorowi bazy danych. Zatrudnia i zwalnia pracowników oraz ma możliwość korygowania błędów.

Podczas wywoływania swoich procedur korzysta z parametrów : @prac - do funkcjonalności dla zwykłego pracownika , @zesp - jest to parametr typu Integer oraz oznacza indeks zespołu , @cz - jest to parametr typu Integer oraz oznacza indeks członka zespołu, @wyn - Nie ujemny integer określający wysokośc wynagrodzenia dla poszczególnego członka , @czas - Nie ujemny integer określający czas pracy , @kl - Nie ujemny integer określający id klienta, @nazwa

- varchar(90) okreslajacy nazwe , @wart - Integer określający wartość , @zl - nie ujemny integer określający id zlecenia , @pr - nie ujemny integer określający id projektu , @dz - nie ujemny integer określający id działu, @st - varchar(90) określający status projektu , @atr - varchar(90) określający atrybut w tabeli, @nowy – varchar(90) określający nową wartość atrybutu. **MOŻE USTALIĆ WARTOŚĆ KAŻDEGO PARAMETRU !**

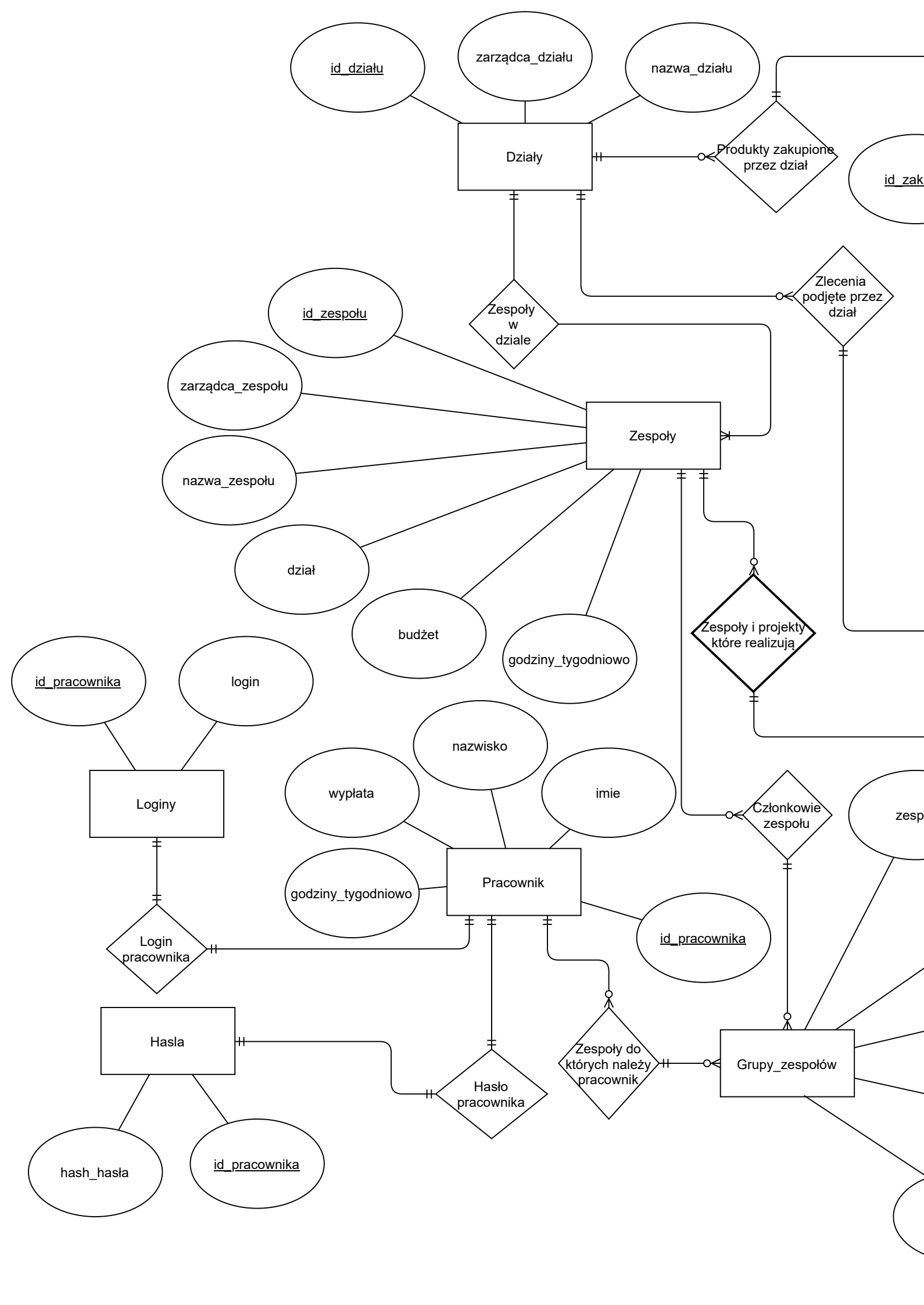
- Korekta_danych(@prac,@atr, @nowy) : Procedura modyfikuje wartości w tabeli Pracownik w miejscu gdzie "id_pracownika" = @prac . Procedura modyfikuje atrybut @atr.
- Korekta_historii(@prac,@zesp,@atr, @nowy) : Procedura modyfikuje wartości w tabeli Grupy_zespołów w miejscu gdzie "pracownik" = @prac oraz "zespol" = @zesp, modyfikuje atrybut @art (zazwyczaj powinien być to status) istnieje tez możliwośc wpisania w miejsce @atr "Delete" wtedy procedura usuwa wiersz a tabelii.
- Dodaj_dział(@prac, @nazwa) : Procedura dodaje do tabeli Działy nowy dział , gdzie wartośc atrybutu zarządca_działu = @prac oraz nazwa_działu = @nazwa.
- Zwolnij_pracownika(@prac) : Procedura usuwa z tabel Loginy i Hasła krotki gdzie id_pracownika = @prac . Ponadto zmienia status pracownika o indeksie @prac we wszystkich projektach na "Były pracownik".
- ZM_zarządcy_działu(@dz , @prac) : Procedura modyfikuje tabele Działy zmieniając atrubut zarządca_działu na @prac w miejscu gdzie id_działu = @dz.
- Przeniesienie_zlecenia(@zl, @dz) : Procedura działa transakcyjnie, zmiania status wszystkich projektów związanych z zleceniem @zl na "Zamkniety" oraz redukuje ich budżety do zera. Nastepnie modyfikuje tabele Zlecenia gdzie id_zlecenia = @zl zmieniając wartość dzial na @dz.
- Zamknij_dział(@dz) : Procedura działa transakcyjnie , wywołuje procedure Przeniesienie_zlecenia dla każdego zlecenia w dziale @dz (podczas uruchomienia wyświetla się okno pytające do , którego działu przenieśc zlecenie). Następnie usuwa wszystkich pracowników z zespołów które należą do @dz (Szef może wywoływać procedury zarządcy zespołu). Następnie zmienia zarządce działu na 0.

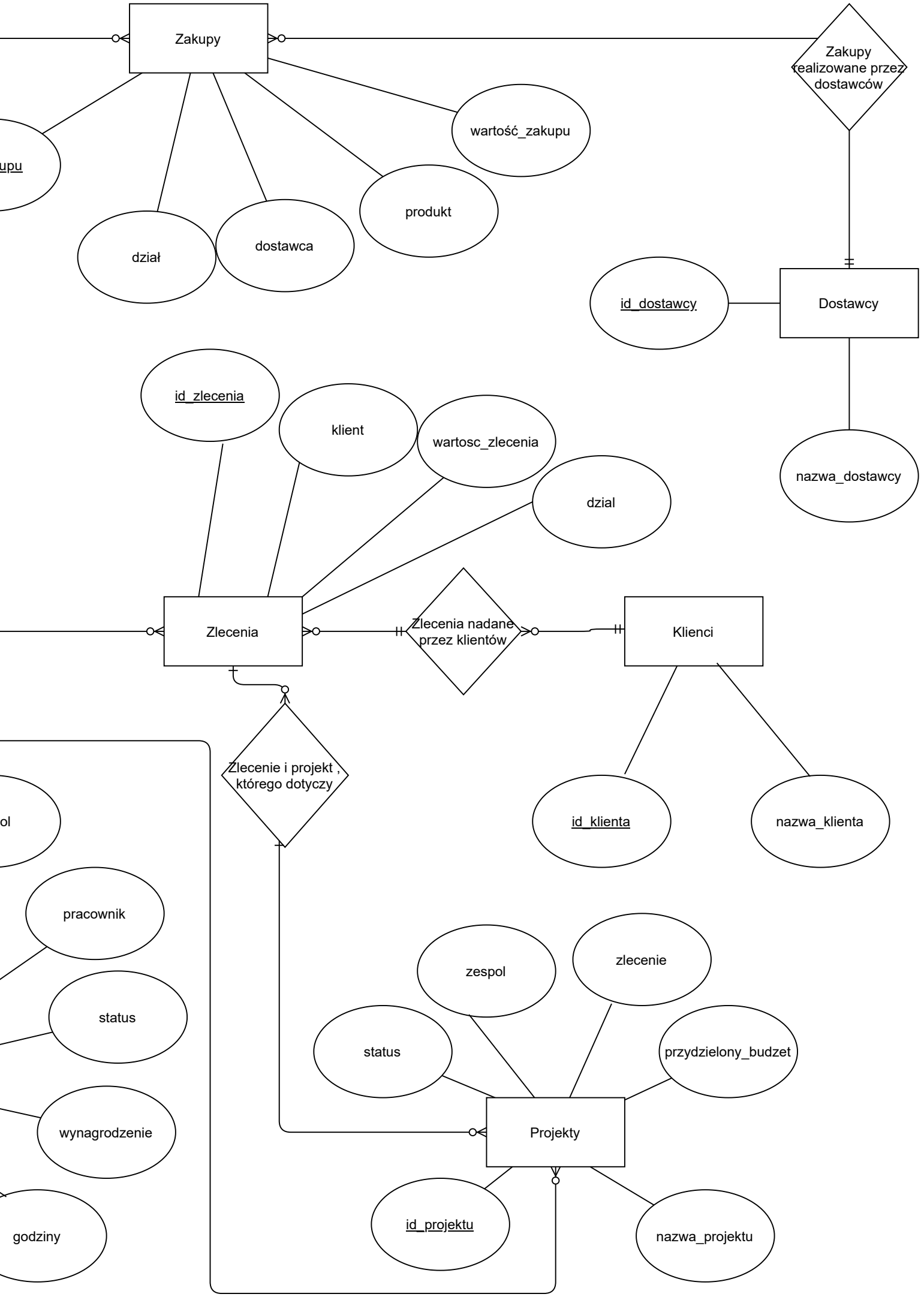
Triggery:

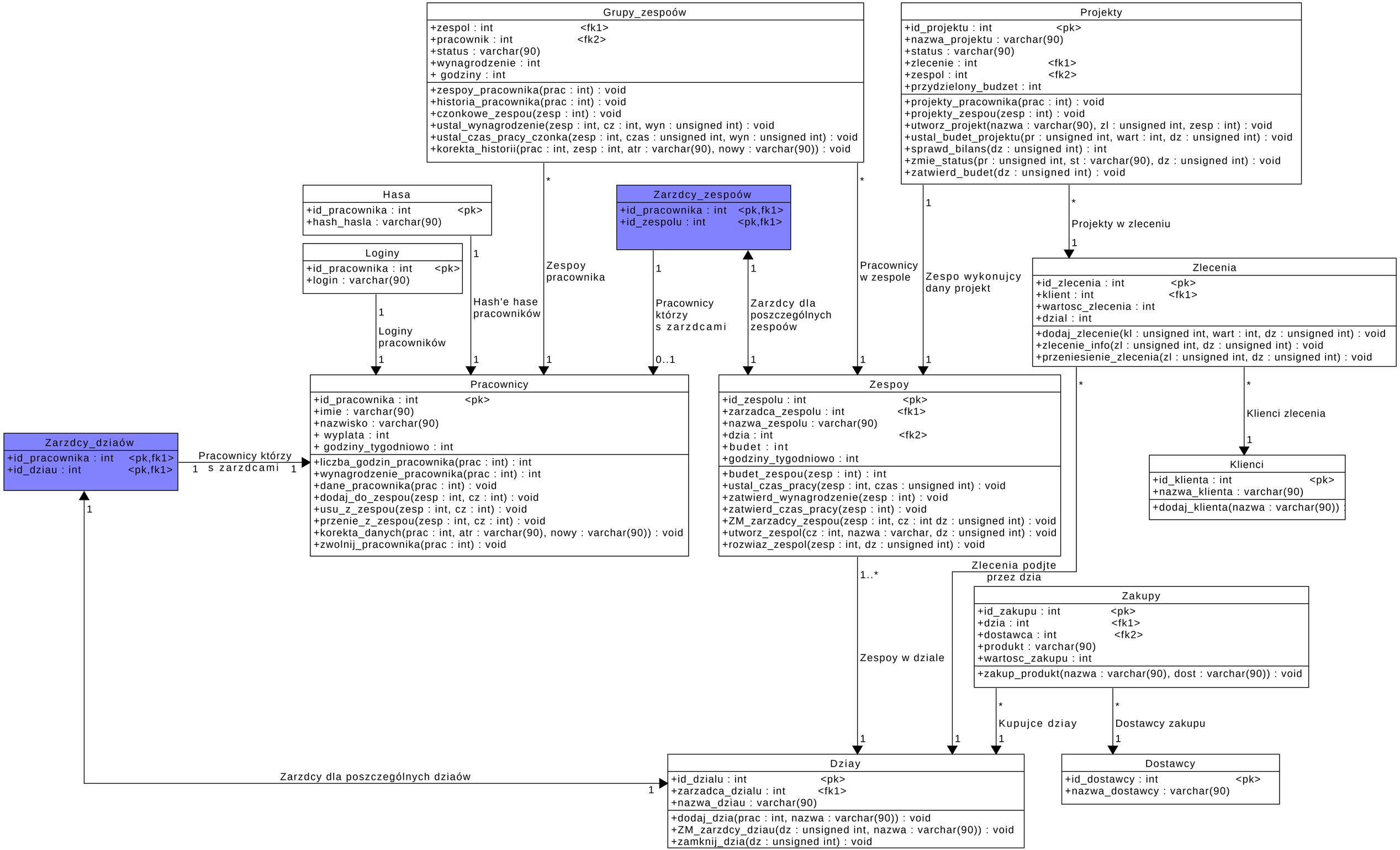
- Działy:
  - update kolumny zarządca_działu powoduje odpowiednią zmianę w widoku zarządcy_działów i tabeli grupy_zespołów
- Zespoły:
  - update kolumny zarządca_zespołu powoduje odpowiednią zmianę w widoku zarządcy_zespołów
  - update kolumny budżet zmniejsza wynagrodzenia pracowników z tabeli grupy_zespołów proporcjonalnie do ich wysokości w przypadku, gdy ich suma przekracza nową wartość kolumny tak aby suma wynagrodzeń nigdy nie przekraczała budżetu
  - update kolumny godziny_tygodniowo zmienia proporcjonalnie liczbę godzin pracowników z tabeli grupy_zespołów tak by ich suma zawsze była równa wartości tej kolumny
- Grupy_zespołów:
  - update kolumny godziny powoduje odpowiednią zmianę w kolumnie godziny_tygodniowo
  - update kolumny status powoduje odjęcie wkładu kolumn wynagrodzenie i godziny z tej tabeli w kolumnach wypłata i godziny_tygodniowo z tabeli Pracownicy
- Zlecenia:
  - usunięcie ostatniego rekordu, który jest powiązany z danym klientem powoduje usunięcie tego klienta
- Zakupy:
  - usunięcie ostatniego rekordu powiązanego z danym dostawcą powoduje usunięcie tego dostawcy

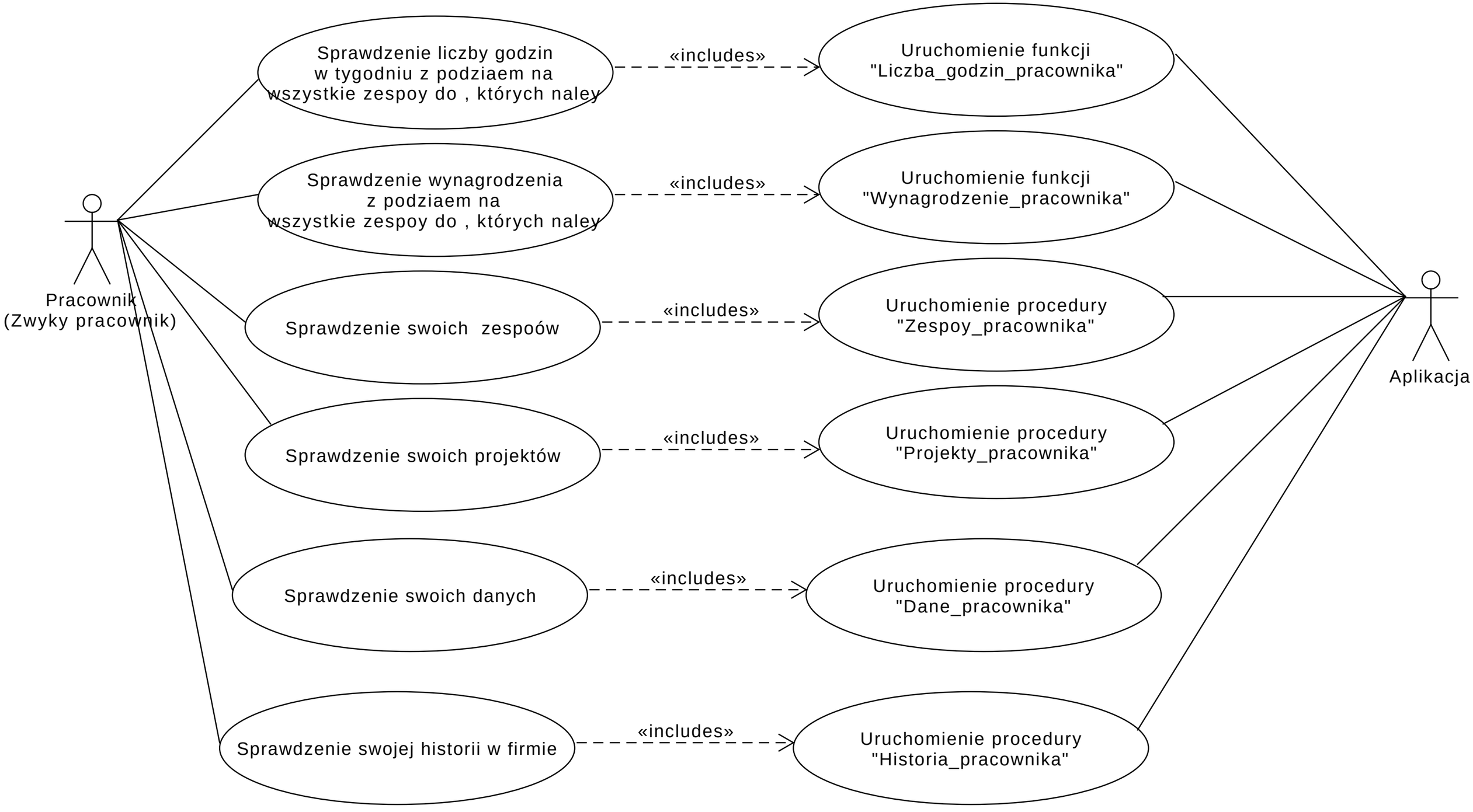
Poziomy dostępu:

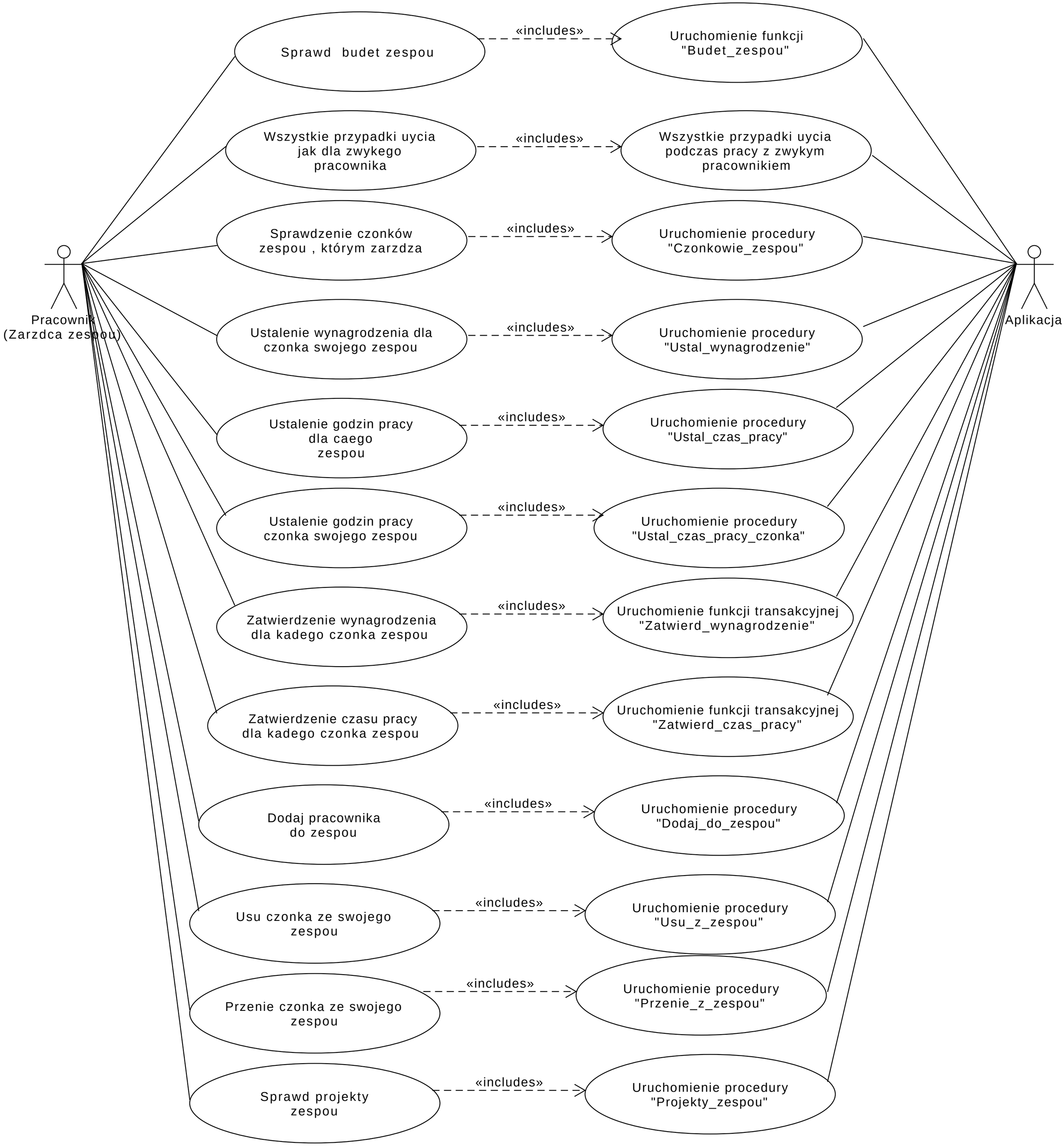
- Pracownik:
  Dostęp do Selectowania tabel: Pracownicy, Grupy_zespołów, Zespoły, Projekty
- Zarządca zespołu:
  To co wyżej+
  Dostęp do Updatowania tabel: Grupy_zespołów, Zespoły
- Zarządca działu:
  To co wyżej+
  Dostęp do Selectowania tabel: Działy, Projekty, Zlecenia, Klienci, Zakupy, Dostawcy
  Dostęp do Updatowania tabel: Projekty, Zlecenia, Klienci, Zakupy, Dostawcy
- Szef:
  Pełny dostęp.


id_działu
zarządca_działu
nazwa_działu
Działy
Produkty zakupione przez dział
id_zak
Zlecenia podjęte przez dział
id_zespołu
Zespoły w dziale
zarządca_zespołu
Zespoły
nazwa_zespołu
dział
budżet
godziny_tygodniowo
Zespoły i projekty które realizują
id_pracownika
login
Loginy
nazwisko
wypłata
imie
Członkowie zespołu
zesp
godziny_tygodniowo
Pracownik
id_pracownika
Login pracownika
Hasla
Zespoły do których należy pracownik
Grupy_zespołów
Hasło pracownika
hash_hasła
id_pracownika


Zakupy
Zakupy realizowane przez dostawców
wartość_zakupu
produkt
dostawca
dział
id_dostawcy
Dostawcy
nazwa_dostawcy
id_zlecenia
klient
wartosc_zlecenia
dzial
Zlecenia
Zlecenia nadane przez klientów
Klienci
Zlecenie i projekt , którego dotyczy
id_klienta
nazwa_klienta
pracownik
status
wynagrodzenie
godziny
zespol
zlecenie
status
przydzielony_budzet
Projekty
id_projektu
nazwa_projektu


Grupy_zespoów
+zespol : int <fk1>
+pracownik : int <fk2>
+status : varchar(90)
+wynagrodzenie : int
+ godziny : int
+zespoy_pracownika(prac : int) : void
+historia_pracownika(prac : int) : void
+czonkowe_zespou(zesp : int) : void
+ustal_wynagrodzenie(zesp : int, cz : int, wyn : unsigned int) : void
+ustal_czas_pracy_czonka(zesp : int, czas : unsigned int, wyn : unsigned int) : void
+korekta_historii(prac : int, zesp : int, atr : varchar(90), nowy : varchar(90)) : void
Projekty
+id_projektu : int <pk>
+nazwa_projektu : varchar(90)
+status : varchar(90)
+zlecenie : int <fk1>
+zespol : int <fk2>
+przydzielony_budzet : int
+projekty_pracownika(prac : int) : void
+projekty_zespou(zesp : int) : void
+utworz_projekt(nazwa : varchar(90), zl : unsigned int, zesp : int) : void
+ustal_budet_projektu(pr : unsigned int, wart : int, dz : unsigned int) : void
+sprawd_bilans(dz : unsigned int) : int
+zmie_status(pr : unsigned int, st : varchar(90), dz : unsigned int) : void
+zatwierd_budet(dz : unsigned int) : void
Hasa
+id_pracownika : int <pk>
+hash_hasla : varchar(90)
Loginy
+id_pracownika : int <pk>
+login : varchar(90)
Zarzdcy_zespoów
+id_pracownika : int <pk,fk1>
+id_zespolu : int <pk,fk1>
Zlecenia
+id_zlecenia : int <pk>
+klient : int <fk1>
+wartosc_zlecenia : int
+dzial : int
+dodaj_zlecenie(kl : unsigned int, wart : int, dz : unsigned int) : void
+zlecenie_info(zl : unsigned int, dz : unsigned int) : void
+przeniesienie_zlecenia(zl : unsigned int, dz : unsigned int) : void
Pracownicy
+id_pracownika : int <pk>
+imie : varchar(90)
+nazwisko : varchar(90)
+ wyplata : int
+ godziny_tygodniowo : int
+liczba_godzin_pracownika(prac : int) : int
+wynagrodzenie_pracownika(prac : int) : int
+dane_pracownika(prac : int) : void
+dodaj_do_zespou(zesp : int, cz : int) : void
+usu_z_zespou(zesp : int, cz : int) : void
+przenie_z_zespou(zesp : int, cz : int) : void
+korekta_danych(prac : int, atr : varchar(90), nowy : varchar(90)) : void
+zwolnij_pracownika(prac : int) : void
Zespoy
+id_zespolu : int <pk>
+zarzadca_zespolu : int <fk1>
+nazwa_zespolu : varchar(90)
+dzia : int <fk2>
+budet : int
+godziny_tygodniowo : int
+budet_zespou(zesp : int) : int
+ustal_czas_pracy(zesp : int, czas : unsigned int) : void
+zatwierd_wynagrodzenie(zesp : int) : void
+zatwierd_czas_pracy(zesp : int) : void
+ZM_zarzadcy_zespou(zesp : int, cz : int dz : unsigned int) : void
+utworz_zespol(cz : int, nazwa : varchar, dz : unsigned int) : void
+rozwiaz_zespol(zesp : int, dz : unsigned int) : void
Klienci
+id_klienta : int <pk>
+nazwa_klienta : varchar(90)
+dodaj_klienta(nazwa : varchar(90))
Zarzdcy_dziaów
+id_pracownika : int <pk,fk1>
+id_dziau : int <pk,fk1>
Zakupy
+id_zakupu : int <pk>
+dzia : int <fk1>
+dostawca : int <fk2>
+produkt : varchar(90)
+wartosc_zakupu : int
+zakup_produkt(nazwa : varchar(90), dost : varchar(90)) : void
Dziay
+id_dzialu : int <pk>
+zarzadca_dzialu : int <fk1>
+nazwa_dziau : varchar(90)
+dodaj_dzia(prac : int, nazwa : varchar(90)) : void
+ZM_zarzdcy_dziau(dz : unsigned int, nazwa : varchar(90)) : void
+zamknij_dzia(dz : unsigned int) : void
Dostawcy
+id_dostawcy : int <pk>
+nazwa_dostawcy : varchar(90)
Zespoy pracownika
Hash'e hase pracowników
Loginy pracowników
Pracownicy którzy s zarzdcami
Zarzdcy dla poszczególnych zespoów
Pracownicy w zespole
Zespo wykonujcy dany projekt
Projekty w zleceniu
Klienci zlecenia
Zlecenia podjte przez dzia
Zespoy w dziale
Kupujce dziay
Dostawcy zakupu
Zarzdcy dla poszczególnych dziaów


Pracownik
(Zwyky pracownik)
Sprawdzenie liczby godzin
w tygodniu z podziaem na
wszystkie zespoy do , których naley
Sprawdzenie wynagrodzenia
z podziaem na
wszystkie zespoy do , których naley
Sprawdzenie swoich zespoów
Sprawdzenie swoich projektów
Sprawdzenie swoich danych
Sprawdzenie swojej historii w firmie
«includes»
«includes»
«includes»
«includes»
«includes»
Uruchomienie funkcji
"Liczba_godzin_pracownika"
Uruchomienie funkcji
"Wynagrodzenie_pracownika"
Uruchomienie procedury
"Zespoy_pracownika"
Uruchomienie procedury
"Projekty_pracownika"
Uruchomienie procedury
"Dane_pracownika"
Uruchomienie procedury
"Historia_pracownika"
Aplikacja


Pracownik
(Zarzdca zespou)
Aplikacja
Sprawd budet zespou
«includes»
Uruchomienie funkcji
"Budet_zespou"
Wszystkie przypadki uycia
jak dla zwykego
pracownika
«includes»
Wszystkie przypadki uycia
podczas pracy z zwykym
pracownikiem
Sprawdzenie czonków
zespou , którym zarzdza
«includes»
Uruchomienie procedury
"Czonkowie_zespou"
Ustalenie wynagrodzenia dla
czonka swojego zespou
«includes»
Uruchomienie procedury
"Ustal_wynagrodzenie"
Ustalenie godzin pracy
dla caego
zespou
«includes»
Uruchomienie procedury
"Ustal_czas_pracy"
Ustalenie godzin pracy
czonka swojego zespou
«includes»
Uruchomienie procedury
"Ustal_czas_pracy_czonka"
Zatwierdzenie wynagrodzenia
dla kadego czonka zespou
«includes»
Uruchomienie funkcji transakcyjnej
"Zatwierd_wynagrodzenie"
Zatwierdzenie czasu pracy
dla kadego czonka zespou
«includes»
Uruchomienie funkcji transakcyjnej
"Zatwierd_czas_pracy"
Dodaj pracownika
do zespou
«includes»
Uruchomienie procedury
"Dodaj_do_zespou"
Usu czonka ze swojego
zespou
«includes»
Uruchomienie procedury
"Usu_z_zespou"
Przenie czonka ze swojego
zespou
«includes»
Uruchomienie procedury
"Przenie_z_zespou"
Sprawd projekty
zespou
«includes»
Uruchomienie procedury
"Projekty_zespou"

Zarzdca dziau

Aplikacja

Zakup nowego produktu «includes» Uruchomienie procedury "Zakup_produkt"

Wszystkie przypadki uycia jak dla zarzdcy zespou z moliwoci wybrania zespou «includes» Wszystkie przypadki uycia podczas pracy z zarzdc pewnego zespou

Podpisywanie umowy z nowym klientem «includes» Uruchomienie procedury "Dodaj_klienta"

Przyjmowanie zlece od klienta «includes» Uruchomienie procedury "Dodaj_zlecenie"

Utworzenie projektu «includes» Uruchomienie procedury "Utworz_projekt"

Ustalenie budetu dla projektu «includes» Uruchomienie procedury "Ustal_budet_projektu"

Sprawd bilans dziau «includes» Uruchomienie funkcji "Sprawd_bilans"

Sprawd informacje o zleceniu «includes» Uruchomienie procedury "Zlecenie_info"

Zmiana statusu projektu «includes» Uruchomienie procedury "Zmie_status"

Zatwierd budety dla zespoów dziau «includes» Uruchomienie funkcji transakcyjnej "Zatwierd_budet"

Zmie zarzdce zespou «includes» Uruchomienie funkcji transakcyjnej "ZM_zarzadcy_zespou"

Utwórz zespó «includes» Uruchomienie procedury "Utworz_zespol"

Rozwi zespó «includes» Uruchomienie procedury "Rozwiaz_zespol"


Szef firmy
Aplikacja
Wszystkie przypadki uycia
jak dla zarzdcy dziau
z moliwoci wybrania dziau
«includes»
Wszystkie przypadki uycia
podczas pracy z zarzdc
pewnego dziau
Modyfikowanie danych pracownika
«includes»
Uruchomienie procedury
"Korekta_danych"
Wgld w dane pracownika
«includes»
Uruchomienie procedur odpowiedzialnych
za wywietlanie informacji o pracowniku
z moliwoci wyboru pracownika
Modyfikowanie historii pracownika
«includes»
Uruchomienie procedury
"Korekta_historii"
Dodawanie nowych dziaów
«includes»
Uruchomienie procedury
"Dodaj_dzia"
Zwalnianie pracownika
«includes»
Uruchomienie procedury
"Zwolnij_pracownika"
Zmiana zarzdcy dziau
«includes»
Uruchomienie procedury
"ZM_zarzdcy_dziau"
Przenoszenie zlecenia do innego dziau
«includes»
Uruchomienie procedury
"Przeniesienie_zlecenia"
Zamkniecie dziau
«includes»
Uruchomienie procedury
"Zamknij_dzia"