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, @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 ,@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.
* 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. **MOŻE USTALIĆ WARTOŚĆ KAŻDEGO PARAMETRU !**

* Korekta\_danych(@prac,@atr) : Procedura modyfikuje wartości w tabeli Pracownik w miejscu gdzie "id\_pracownika" = @prac . Procedura modyfikuje atrybut @atr.
* Korekta\_historii(@prac,@zesp,@atr) : 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.