# Liga Piłkarska

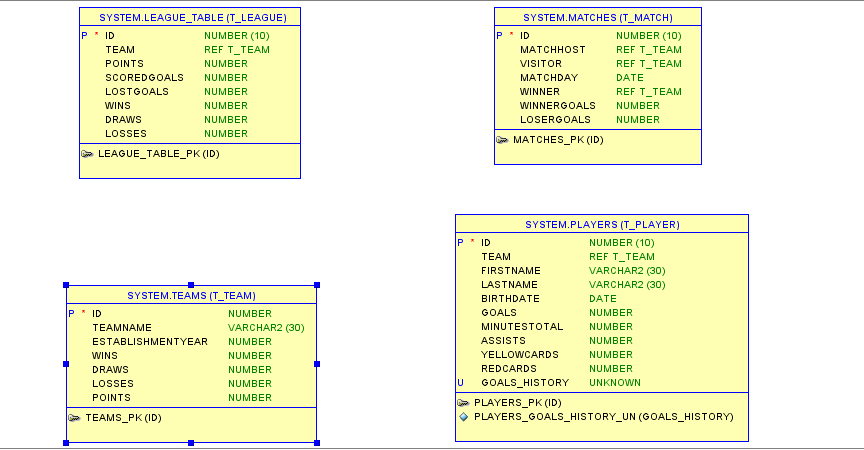
Wybranym przeze mnie projektem było zaprojektowanie tabeli ligi piłkarskiej. Zdecydowałem się na realizację takiego tematu, ponieważ w świetny sposób oddaje on zastosowanie relacyjnych baz danych. Przechowywanie danych na temat ligi piłkarskiej daje duże pole do popisu, jednak w projekcie zdecydowałem się na oddanie najważniejszych takich jak:

* Dane dotyczące meczu
* Tabela ligowa
* Dane dotyczące zawodników
* Dane dotyczące drużyny

Główne założenia, przy realizacji tematu były:

* Baza danych obsługuje tylko jedną ligę piłkarską
* Baza danych obsługuje tylko jeden sezon ligowy
* Baza danych przechowuje historię przeprowadzonych meczów
* Baza danych przechowuje drużyny
* Baza danych przechowuje zawodników
* Baza danych zawiera tabelę ligową
* Tabela ligowa składa się z kolumn, z których każda zawiera: drużynę, liczbę punktów, liczbę zwycięstw, liczbę remisów, liczbę porażek, gdzie liczba punktów wyliczana jest na podstawie zwycięstw oraz remisów
* Dane tabeli są aktualizowane automatycznie po wprowadzeniu wyniku meczu
* System posiada możliwość wyświetlania najlepszych strzelców ligi
* System posiada możliwość wyświetlania najlepszych asystentów ligi
* System posiada możliwość wyświetlania tabeli ligowej w postaci posortowanej
* Drużyna może rozgrywać maksymalnie jeden mecz w podanym terminie
* Zawodnik może być przypisany do maksymalnie jednej drużyny

## Typy określone w projekcie



Do zaimplementowania systemu wykorzystano

4 typy:

* **t\_league** – określający ligę
* **t\_match** – określający mecz
* **t\_team** – określający drużynę
* **t\_player** – określający zawodnika

4 pakiety:

* **league\_utils** – zawierający procedury oraz wyjątki potrzebne do obsługi ligi
* **team\_utils** – zawierający procedury, funkcje oraz wyjątki potrzebne do obsługi drużyn
* **player\_utils** – zawierający procedury oraz wyjątki potrzebne do obsługi zawodników
* **match\_utils** - zawierający procedury oraz wyjątki potrzebne do obsługi meczów

Identyfikatory zawarte w tabelach są określane przez sekwencje, dzięki czemu, pewnym jest, że są one unikalne. Takie zapewnienie pozwala na zastosowanie ich w momencie odwoływania się do konkretnych obiektów i właśnie w taki sposób zostało to zrealizowane w tym projekcie. Ponadto do przechowywania wielu typów jednego rodzaju w innym typie, zastosowano **nested table** (goals\_history w t\_player), natomiast do informowania użytkownika o liderze tabeli przy każdej zmianie jej wartości posłużył **TRIGGER**. W projekcie zawarte są również utworzone, własne wyjątki (**EXCEPTION**), takie jak np. PLAYER\_ALREADY\_EXISTS, które pomagają w realizacji logiki procedur oraz funkcji.