**Математички факултет у Београду**

**Мастер рад**

# Моделовање турнирских такмичења помоћу Oracle софтверских алата

**Студент: Душа Вуковић**

**Ментор: проф. др. Душан Тошић**

27. јун 2012. године

*Апстракт*

У раду су, кроз пример моделовања карате-турнирских такмичења, приказане специфичности моделовања употребом Oracle софтверских алата. Основа за добру базу података је добар логички модел на основу којег се гради база, а затим и апликација за крајњег корисника. Сви пословни захтеви се приказују на дијаграму ентитета и веза (скраћено ЕРД од енгл. ERD - Entity Relationship Diagram) и формира се пратећа документација. Алат који се користи за моделовање је *Oracle Data Modeler*. Сама база може да се изгради у софтверском алату *Oracle Database 10g Express Edition* доступном на сајту www.oracle.com. У раду је описан поступак пресликавања конкретног модела у скуп табела коришћењем овог алата. Да би слика о Oracle-окружењу била комплетна, приказано је како се креирана база података може користити постављањем упита коришћењем језика SQL, односно, PL/SQL. Овај рад се ослања на курс Дизајн база података и програмирање у језику SQL (енгл. Database design and programming with SQL) који је део обука које спроводи Академија Oracle и које су приказане на [1].

# САДРЖАЈ

# 1. Увод

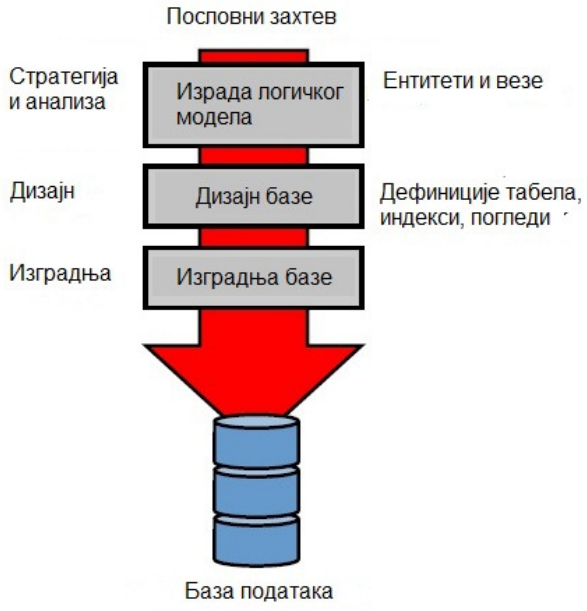
Модерни системи за рад са базама података пружају много више од механизма за чување података. Људи су и пре појаве рачунара били у могућности да чувају огромне количине података. Можемо да узмемо за пример било коју библиотеку. Оно што су нам донели модерни системи је ефикасност у претраживању података и добијању жељених информација из њих. Замислимо да не постоји систем за управљањем базом података за неко турнирско такмичење. Сви подаци о такмичарима и свим одржаним мечевима се бележе на папирима. Уколико нам треба статистика једног појединачног меча (списак освојених поена, број додељених казни, интервенције лекара), потребно је да међу папирима пронађемо записник са тог меча и прочитамо податке о том мечу. Међутим, ситуација је компликованија ако нам је потребна нека информација коју можемо да добијемо само на основу података о свим мечевима. Рецимо да нам је важно да видимо која од техника ја била најзаступљенија у мечевима (ударац руком у тело, ударац руком у главу, ударац ногом...) да би тренери знали на шта да се концентришу на тренинзима. Или, колико је укупно додељено казни због повреде противника, да би се више вежбало на прецизности и контроли удараца. Без рачунара, долазак до ових информација је дуг и мукотрпан посао који подразумева да се прелистају записници са свих мечева и да се ручно издвајају и бележе подаци који нас интересују. Са системом за управљање базама података до ових, и многих других, информација можемо да дођемо исписивањем једноставног упита и његовим извршењем за делић секунде.

Како су рачунари постајали доступни, прво фирмама, па касније и појединцима, тако се развијао и концепт система за управљање подацима. Едгар Ф. Код, Енглески научник, је око 1970. године развио релациони модел за управљање базама података и тиме поставио теоријску основу релационих база података које се и данас користе. Године 1976. Амерички научник, Питер Чен, је увео модел ентитета и веза за дизајн база података. Модел ентитета и веза је логички модел, пројекат који је независан од каснијег избора система за управљање базамам података помоћу којег ће се креирати база. Раних осамдесетих година двадесетог века се појавио комерцијално доступан систем за базе податак: Oracle, верзија 2. Средином осамдесетих језик упита SQL (енгл. Structured Query Language) постаје стандард као језик за рад са подацима у бази података.

Поступак креирања базе података је сложен и састоји се из више фаза. Прва фаза подразумева истраживање самог пословања за које је потребно креирати базу. У овом случају то је организација турнирског такмичења неке од борилачких вештина. Током ове фазе, будући корисници сарађују са стручњацима који развијају базу и обезбеђују им све потребне информације. Најважније је да се разуме како функционише организација турнира. Може да се уочи да су битни актери током турнира: такмичари, судије, тренери, лекари, записничари, клубови... Уочавају се и остали битни елементи турнира, као што су појединачни или екипни мечеви или евентуални преглед код лекара. Бележе се подаци које морамо да чувамо у бази као што су: назив, адреса, матични број за клуб или име, презиме и број лиценце за сваког судију. Уочавају се везе међу подацима и остала правила пословања којима се описује како изгледа пријава клубова и такмичара на турнир, распоред судија по борилиштима, процедуре у случају повреда, проглашење победника меча и остало. Стручњаци који креирају базу праве логички модел будуће базе који се назива **дијаграм ентитета и веза** (скраћено ЕРД или ЕР дијаграм, од енгл. ERD – Entity Relationship Diagram). За креирање дијаграма може да се користи софтвер *Oracle Data Modeler*([2]). Поред дијаграма се креира и додатна документација која ће бити важна касније приликом изградње базе. Будући корисници и стучњаци интензивно сарађују током ове фазе да би модел био што бољи и у потпуности одговарао потребама пословања. Ова фаза се назива и **фазом стратегије и анализе**. Током следеће фазе, која се назива **фаза дизајна**, се врше припреме за изградњу базе тако што се на основу креираног логичког модела уочавају

потребне табеле, кључеви, типови података, индекси и остало. Завршна фаза је **фаза изградње** базе током које се, на основу извршених припрема, физички креира база са свим потребним табелама и осталим објектима. Након тестирања, база може да се пусти у рад и пуни подацима. Креирање табела и осталих објеката које база мора да има, као и пуњење подацима, се обавља уз помоћ софтвера *Oracle Database 10g Express Edition* који може да се преузме са [2], и језика упита специјализованог за управљање подацима SQL. Овај упитни језик се користи и за добијање потребних информација из података који су сачувани у бази. Поред језика SQL користи се и програмски језик PL/SQL у којем се користе наредбе језика SQL, а који још има могућност рада са променљивима, управљање током програма (наредбе гранања и циклуса), као и писање процедура и функција.

Кроз овај рад ће бити приказан поступак моделовања и креирања базе података за турнирко такмичење. На следећој слици је урађена систематизација управо описаних фаза потребних да се дође до готове и оперативне базе података према [1].



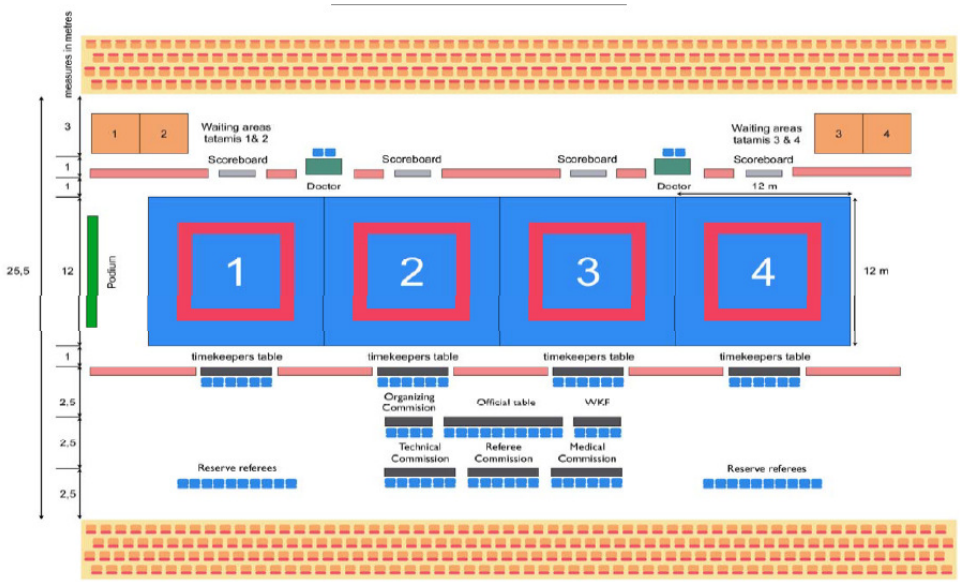

**Слика 1. Процес креирања базе података**

## 2. Турнирско такмичење

Први корак ка креирању базе података је упознавање са детаљима организације турнирских такмичења, потребним подацима који се бележе и информацијама које су потребне за побољшање пословања.

Овде ће посебна пажња бити посвећена моделовању карате-турнира, али исти принципи се користе и приликом моделовања других врста турнира. Тренирање борилачких спортова подразумева развијање снаге, издржљивости и брзине кроз кондиционе тренинге, савладавање основних ударних и одбамбених техника, усавршавање ката и борбених вештина. Иако се током тренинга ради на свим аспектима спорта, такмичари се углавном опредељују за такмичење или у катама или у борбама. Кате су борбе са замишљеним противницима и кроз њих се демонстрира познавање технике, као и брзина, снага и концентрација. Борба подразумева суочавање два противника у рингу под посебним правилима. Правила укључују дефиниције дозвољених удараца и ударних површина. На пример, ударац отвореном шаком, тј. шамар, није дозвољена техника у борби, а у забрањене ударне површине спадају врат и колено.

Турнирска такмичења борилачких спортова се одржавају у спортским халама. Сама организација простора и целог турнира је углавном уређена према препорукама светских федерација([3]). Мечеви се одржавају на борилиштима. Следећа слика приказује распоред четири борилишта према предлогу Светске карате федерације WKF који је изложен на сајту [3].




Слика 2. Распоред четири борилишта према предлогу Светске карате федерације WKF

Турнир се састоји из великог броја мечева. У оквиру сваког меча, два такмичара се суочавају или у борби, или у извођењу ката. Екипни меч се састоји из више појединачних мечева у којим се на крају саберу резултати. Клубови пријављују такмичаре на турнир. Са такмичарима долазе и њихови тренери. Клубови могу да имају и своје судије. Организатори распоређују судије по борилиштима. Потребно је доста и помоћног особља за вођење записника на борилиштима, као и лекарска екипа. Детаљније ће се размотрити организација турнира само због објашњења процеса моделовања турнира.

# 3. Моделовање

## 3.1. Ентитети

Основни поступак у логичком моделовању је процес препознавања података које је потребно да чувамо и у каквој су међусобној вези ти подаци. Уочавају се ентитети. Под ентитетом се сматра све што је битно за посао и што је дефинисано подацима које треба да чувамо. Такмичаре на турнир пријављује клуб у којем тренирају и у којем су регистровани. Тако да, у оквиру турнирских такмичења можемо, на пример, да уочимо ентитет *Klub*. Ентитете на логичком моделу приказујемо у правоугаонику са заобљеним ћошковима (енгл. Soft-box). Именујемо их у једнини.


Слика 3. Ентитет *Klub*

Сваки ентитет може да има више инстанци. Инстанца је примерак ентитета. Тако је, на пример, Клуб „Чукарица“ једна инстанца ентитета *Klub*.

Унутар ентитета набрајамо атрибуте. Атрибути су подаци које чувамо. За клубове су битни следећи подаци: назив клуба, адреса, број пословног рачуна, број решења, матични број, ПИБ број, телефон, факс, мејл, сајт, адреса објекта где се одвија делатност клуба, адреса за пошту и име лица на које се шаље пошта. Следи слика формулара са листом потребних података о клубовима преузета са сајта Београдског карате савеза [4].

| | | |
|---|---|---|
| | **BEOGRADSKI KARATE SAVEZ** | BEOGRAD, Deligradska 27 / III<br>tel. 2643–735, 3621–361<br>www. beokarate.org.yu<br>E-mail: beokarate@EUnet.yu |
| | upitni list za registraciju i formiranje baze podataka saveza<br>KLUB | Poslovni račun:<br>255-38360101000-41<br>PIB broj : 102957359 |

| | |
|---|---|
| **KARATE KLUB** | |
| Adresa i poštanski broj: | |
| Broj poslovnog računa: | |
| Broj rešenja: | |
| Matični broj: | |
| PIB broj: | |
| Telefon: | Fax: |
| E-mail: | Sajt: www. |
| Adresa objekta gde se odvija delatnost kluba: | |
| Adresa za poštu: | |
| Prezime i ime lica na koje se šalje pošta: | |

Molimo Vas da podatke popunite tačno. Vašim potpisom garantujete i snosite odgovornost za tačnost podataka.

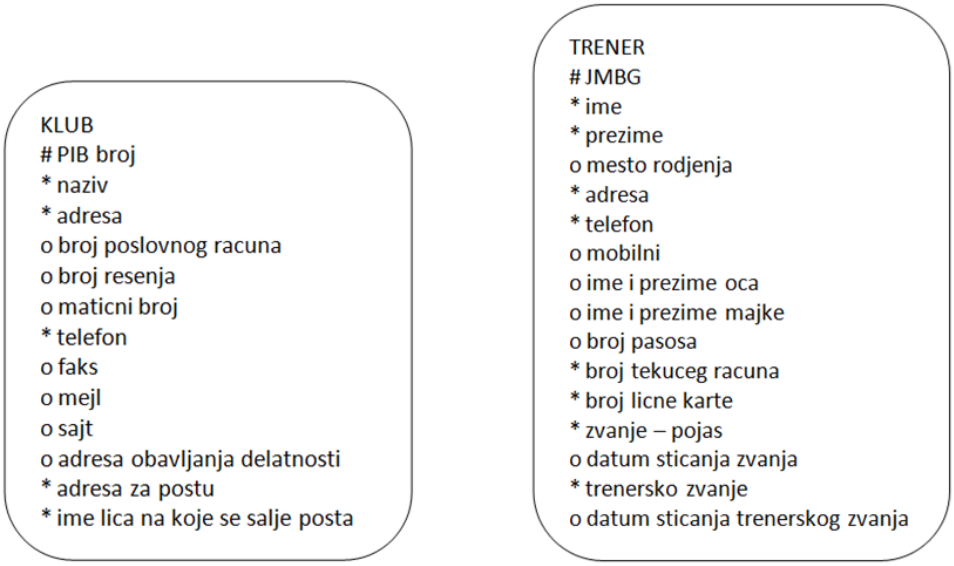
Слика 4. Формулар Београдског карате савеза за регистрацију клуба

Уз сваки од атрибута је потребно навести један од следећа три знака: тарабицу (#), звездицу (*) или кружић (о).

Тарабицом означавамо примарни јединствени идентификатор. То је податак који је јединсвтен за сваку инстанцу ентитета и на основу којег се разликују сваке две инстанце истог ентитета. Може да се деси да постоји и неколико података са овим својством. Онда се један изабере да буде примарни јединствени идентификатор, а остали се сматрају кандидатима за примарни јединствени идентификатор. Примарни јединствени идентификатор може да буде и комбинација два или више атрибута, или да се вештачки уведе ако не постоји природни. У случају клубова, следећи подаци су кандидати за примарни јединствени идентификатор: број пословног рачуна, број решења, матични број и ПИБ број. Порески идентификациони број (ПИБ) је природни примарни јединствени идентификатор јер га пореска управа додељује клубовима, који се третирају као правна лица, у циљу идентификације пореских обвезника и он је јединствен и једини број правног лица за све јавне приходе [5].

Звездицом означавамо атрибут чију вредност морамо да имамо. За такве атрибуте кажемо да су обавезни. На пример, бесмислено је регистровати клуб уколико му не знамо име, али је могуће да у тренутку регистрације клуб нема свој сајт или није суштински важно да се адреса сајта забележи. Примарни јединствени идентификатор је увек обавезан. За атрибуте који нису обавезни кажемо да су опциони и обележавамо их кружићем.

Такмичаре на турнир доводе тренери. У сваком клубу ради један или више тренера. На слици 5 су приказани ентитети *Klub* и *Trener*, са свим потребним атрибутима.

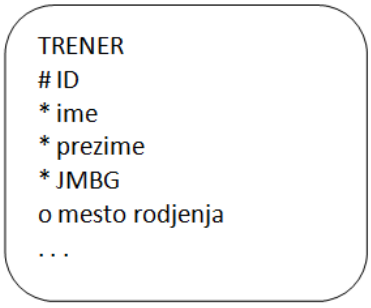



**Слика 5. Ентитети *Klub* и *Trener***

Сваки тренер има ЈМБГ. Јединствени матични број грађана је природни примарни јединствени идентификатор. Кандидати за примарни јединствени идентификатор су још и: број пасоша и број личне карте. Они се не обележавају посебно, али је потребно да се та чињеница документује зато што ти подаци касније захтевају посебну пажњу. Два тренера не могу да имају исти број пасоша или исти број личне карте, док може да се деси да се исто зову и презивају. О опционалности атрибута доносимо одлуку на основу тога који подаци о тренерима ће морати да се уносе (на пример, не можемо да регистујемо тренера уколико му не знамо име и презиме), а који ће моћи да се изоставе (број мобилног телефона уколико тренер не жели да га да, или га нема). На пример, број пасоша би морао да буде опциони податак. У супротном, уколико бисмо

захтевали да се за сваког тренера унесе број пасоша, имали бисмо проблем код оних тренера који, из било којег разлога, немају пасош.

Јединствени матични број грађана је рогобатан податак који се састоји од 13 цифара и није погодан за лако упоређивања и манипулацију. У таквој ситуацији, или у ситуацијама када не постоји природни примарни јединствени идентификатор, треба да се уведе вештачки. Вештачки примарни јединствени идентификатор је најчешће број и представља јединствену шифру сваке инстанце једног ентитета унутар система, тј. базе података. Узимајући то у обзир, на слици 6 је приказан ентитет *Trener* са вештачким примарним јединственим идентификатором (изостављени су атрибути који су исти као на претходном цртежу).




Слика 6. Ентитет *Trener*

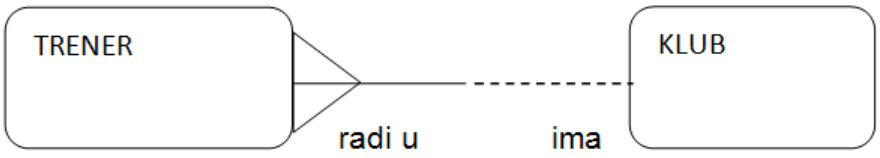
## 3.2. Везе

Ентитети су у природној вези и потребно је уочити, именовати и нацртати све те везе. Само због једноставности цртежа у овом тренутку ће бити изостављени атрибути које сваки ентитет има.

Свака веза повезује тачно два ентитета. Посебан случај чини само рекурзивна веза која иде од ентитета ка истом ентитету и која ће бити објашњена касније.

Свака веза мора да има назив који је описује, као и прецизирану опционалност и кардиналност. Опционалност везе између два ентитета нам говори о томе да ли једна инстанца једног од ентитета мора или не мора да буде у вези са неком инстанцом другог ентитета. Кардиналност (другачије звана степен везе) говори о броју инстанци другог ентитета са којима једна инстанца може да буде у вези. Уколико је веза опциона, линија којом је приказана је испрекидана, у супротном је пуна. Кардиналност је приказана са једном (енгл. Single toe) или више линија (енгл. Crows foot).

Веза између ентитета *Trener* и *Klub* се приказује на следећи начин:




Слика 7. Веза између ентитета *Trener* и *Klub*

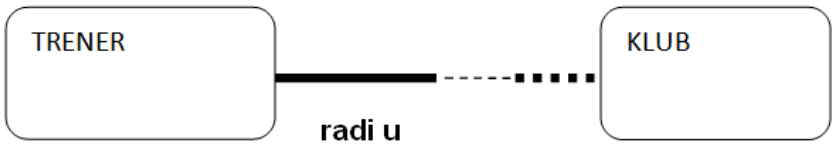
Веза се именује и чита са оба краја, од једног ентитета ка другом и обратно. Приликом читања везе, потребно је нагласити назив везе, кардиналност и опционалност.

Претпоставимо да имамо ентитете А и Б и везу између њих. Веза се чита са две реченице:
Сваки А *може/мора* да *назив везе* са *једним и само једним/једним или више* Б.
Сваки Б *може/мора* да *назив везе* са *једним и само једним/једним или више* А.

Кажемо „може“ ако је веза опциона, а „мора“ уколико је обавезна. Део реченице „једним и само једним“ или „једним или више“ нам показује кардиналност.

Читање веза на овај начин је јако важно јер представља основу добре комуникације између стручњака који креирају модел и будућих корисника за које се креира модел. Реченице су довољно формалне и прецизне за стручњаке који креирају модел, али и довољно блиске обичном говору да би будући корисници могли да их разумеју и помогну у процесу моделовања.

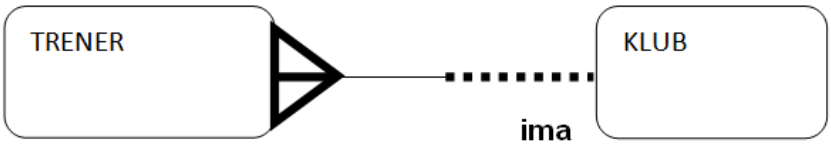
Веза од ентитета *Trener* ка ентитету *Klub* се чита на следећи начин:
Сваки тренер мора да ради у једном и само једном клубу.




Слика 8. Веза од ентитета *Trener* ка ентитету *Klub*

Опционалност везе се чита ближе почетном ентитету (да ли је линија испрекидана или пуна), док се кардиналност чита на удаљеном крају везе, тј. ближе другом ентитету (да ли је једна линија или више). Како је овде линија пуна ближе ентитету *Trener*, тренер мора да ради у неком клубу. Даље од ентитета је само једна линија, па тренер ради у тачно једном клубу.

Веза од ентитета *Klub* ка ентитету *Trener* се чита на следећи начин:
Сваки клуб може да има једног или више тренера.



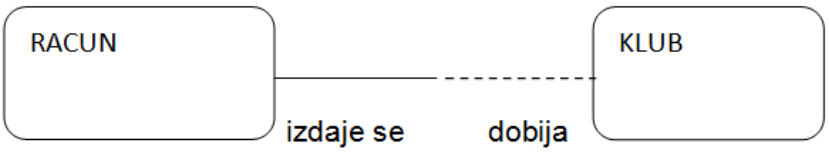

Слика 9. Веза од ентитета *Klub* ка ентитету *Trener*

Линија је испрекидана ближе ентитету *Klub*, па је за клуб опционо да има тренере. Даље од ентитета *Klub* има више од једне линије, па у сваком клубу има један или више тренера.

Опционалност ове везе са оба краја се види у анализи процеса пријаве на турнир. На турнир се прво пријављује клуб. У тренутку док се уносе подаци о клубу, још увек нису познати подаци о његовим тренерима, па је јасно да имамо реалну ситуацију у којој имамо клуб без тренера. Када се уносе подаци о тренеру, подаци о клубу већ постоје, и за тренера мора да се зна ком клубу припада. Из тог разлога је веза обавезна на крају ближем ентитету *Trener*.

Према кардиналности везе се деле на:
1:1 (један према један) – кардиналност је један на оба краја
1:M (један према више) – на једном крају је један, а на другом више
M:M (више према више) – више на оба краја

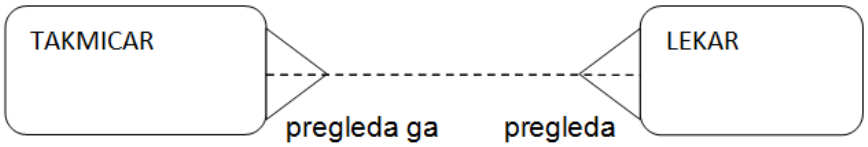
Везе типа 1:1 су ретке, али могу да постоје. На пример, када се клуб пријави на турнир мора да плати котизацију за учешће. Приликом плаћања се издаје рачун са подацима о износу новца, датумом, потписом особе која је примила новац. Рачун са подацима који га описују чини ентитет. Сваки клуб може да добије један и тачно један рачун. Сваки рачун мора да се изда тачно једном клубу. Ова веза је приказана на следећој слици:




Слика 10. Веза између ентитета *Racun* и *Klub*

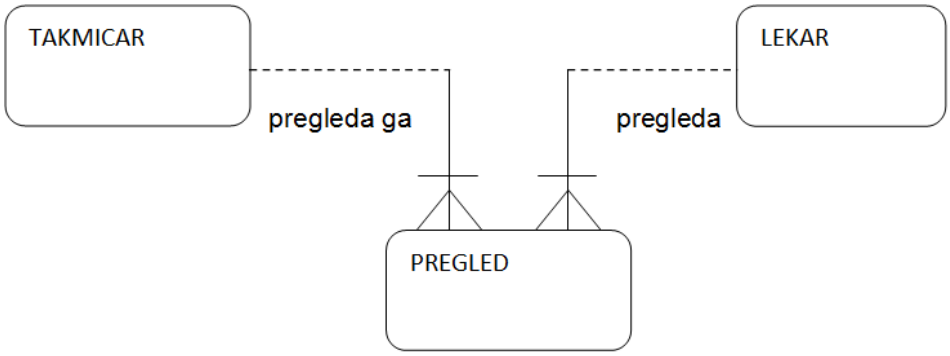
Везе типа 1:M су најчешће и пожељне. Пример такве везе је већ описана веза између ентитета *Klub* и *Trener*.

Везе типа M:M није могуће прилагодити релационом моделу база, и потребно их је модификовати. На турнирима борилачких спортова увек имамо дежурног бар једног, а понекад два или три спортска лекара који реагују и прегледају такмичара уколико дође до неке повреде током меча. Исти такмичар може да буде прегледан од стране лекара неколико пута, а лекар током турнира прегледа више различитих такмичара. Уколико нико није био повређен током такмичења, неће доћи до прегледа тако да је ова веза опциона. Везу приказујемо следећом сликом:




Слика 11. Веза између ентитета *Takmicar* и *Lekar*

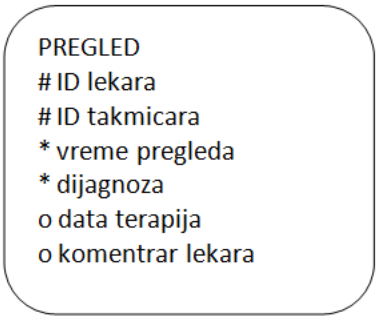
Веза типа M:M крије додатни ентитет који често има и своје атрибуте. Нови ентитет се додаје, као и две везе које га спајају са оба ентитета. Почетна веза се брише. Нова ситуација је приказана на следећој слици. Ентитет *Pregled* је уведен уместо везе M:M која је уочена између ентитета *Lekar* и *Takmicar*.




Слика 12. Ентитет Pregled на месту везе типа M:M између ентитета *Takmicar* и *Lekar*

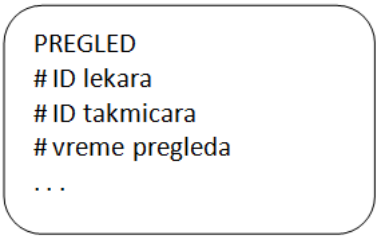
Опционалност почетне везе остаје видљива на крајевима који су ближи ентитетима *Lekar* и *Takmicar*. Такмичар може, а не мора, да доживи повреду и оде на преглед код лекара. Исто тако, лекар може, а не мора, да прегледа једног или више такмичара. Међутим, уколико је дошло до

прегледа, сигурно је дежурни спортски лекар прегледао једног тамичара и веза је обавезна ближе новом ентитету јер су у прегледу обавезно учествовали лекар и такмичар. Две цртице којима су прецртане везе од ентитета *Pregled* ка ентитетима *Takmicar* и *Lekar* наглашавају да је примарни јединствени идентификатор ентитета *Pregled* комбинација примарних јединствених идентификатора ентитета *Lekar* и *Takmicar*. Примарни јединствени идентификатор који се састоји од два или више атрибута се назива сложени или композитни. Ентитет *Pregled* са својим атрибутима је приказан на следећој слици.



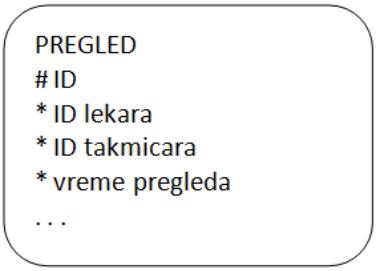

Слика 13. Прва верзија ентитета *Pregled*

Овакав ентитет нам гарантује да је комбинација шифре ентитета *Lekar* и шифре ентитета *Takmicar* јединствена али се појављује проблем у ситуацији када исти такмичар дође на преглед код истог лекара неколико пута током истог или различитих мечева поводом различитих повреда. Да би решили тај проблем, можемо да додамо и време прегледа као део примарног јединственог идентификатора:




Слика 14. Друга верзија ентитета *Pregled*

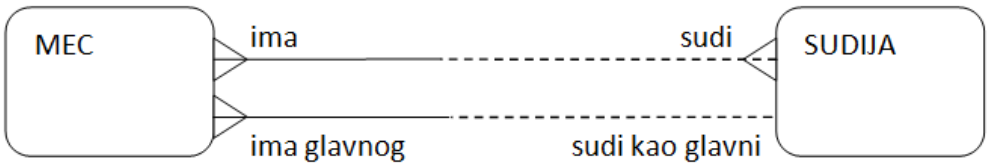
Примарни јединствени идентификатор који се састоји од три атрибута није практичан. Као што је већ раније споменуто, у таквој ситуацији је добро увести нови вештачки примарни јединствени идентификатор.




Слика 15. Коначна верзија ентитета *Pregled*

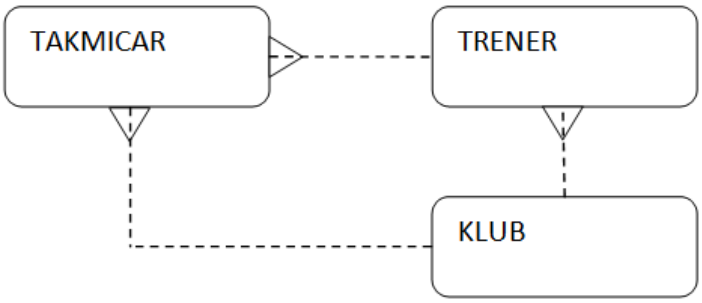
Може да се деси да између два ентитета постоји више од једне везе. Посматрајмо ентитете *Mec* и *Sudija*. На сваком мечу мора да суди више судија. Сваки судија може да суди на једном или

више мечева. Ова веза је типа M:M. За сваки меч се одреди један судија који се прогласи главним судијом за тај меч. Сваки меч има тачно једног главног судију. Исти судија може да буде главни судија у више мечева, па је ова веза типа 1:M. Може да се примети да између ентитета *Mec* и *Sudija* постоје две различите везе.



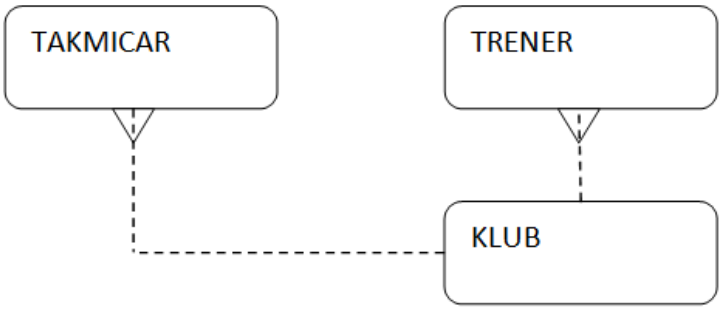

Слика 16. Две различите везе између ентитета *Mec* и *Sudija*

Везе приказане на претходном цртежу треба да остану јер су различите и обе су важне. Другачију ситуацију имамо у следећем примеру.




Слика 17. Дуплиране везе између ентитета *Takmicar*, *Trener* и *Klub*

За сваки клуб се такмичи један или више такмичара, и сваки такмичар се такмичи за тачно један клуб. У сваком клубу ради један или више тренера, и сваки тренер ради у тачно једном клубу. Сваки тренер прати рад једног или више такмичара, и рад сваког такмичара прати тачно један тренер. Ентитети *Takmicar* и *Klub* су повезани и ми можемо да видимо ком клубу припада такмичар. Међутим, то можемо да видимо и на други начин. Сваки такмичар има тренера, а за тренера се зна у ком клубу ради, па самим тим и сви такмичари које он прати припадају истом клубу. Потребно је да везе које се непотребно понављају уклонимо. Уколико остану, доприносе непрегледности, а потенцијално могу да доведу и до грешака. Како за организацију турнира није неопходно да знамо за сваког такмичара име његовог тренера, из ове ситуације је најбоље уклонити везу између ентитета *Takmicar* и *Trener*. У некој другој ситуацији, на пример када бисмо креирали модел за пословање клубова, боље би било да се уклони веза између ентитета *Takmicar* и *Klub* да не бисмо изгубили важан податак о томе за ког такмичара је задужен који тренер. Нова ситуација за модел турнира је приказана на следећој слици.




Слика 18. Везе без понављања између ентитета *Takmicar*, *Trener* и *Klub*

## 3.3. Дијаграм ентитета и веза

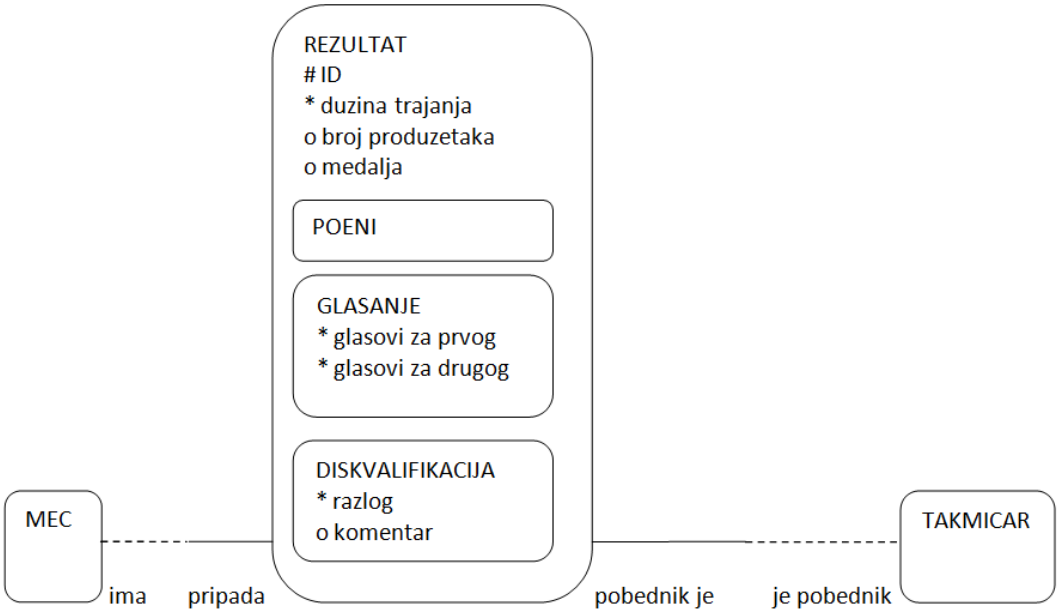
Сви ентитети и везе које их повезују се приказују на дијаграму ентитета и веза названом ЕРД (енгл. ERD – Entity Relationship Diagram)([1]). Дијаграм ентитета и веза се назива и логички модел. Он је потпуно независан од избора софтвера за касније креирање саме базе. Приликом цртања ЕР дијаграма потребно је водити рачуна о његовој прегледности и читљивости. Линије које представљају везе не би требало да се пресецају. Уколико дијаграм има много ентитета, могуће је поделити га на неколико цртежа. Ентитет који учествује у много веза или је јако важан за пословање потребно је поставити на централно место на дијаграму.

Ентитетима и везама су описана структурна пословна правила (енгл. Structural Business Rules). Структурна пословна правила описују податке и везе међу њима. Постоје и друга правила која не могу да се прикажу помоћу ЕР дијаграма. Таква правила називамо поцедуралним пословним правилима (енгл. Procedural Business Rules). Процедурална пословна правила не могу да се нацртају на дијаграму. Њих морамо посебно да документујемо да би у некој каснијој фази изградње целог система била обрађена програмерски. На пример, на дијаграму можемо да нацртамо ентитете *Boriliste*, *Sudija* и *Sudjenje* којима се приказује ситуација да на сваком борилишту имамо судије које суде у мечевима. Процедурално правило које не можемо да нацртамо, а важно је за пословање, је да на једном борилишту у једном тренутку морају да постоје тачно четири помоћне судије и један главни судија. Још један пример процедуралног правила је да само судија који има лиценцу за кате може да суди кате, а судија који има лиценцу за борбе може да суди борбе. У процедурална правила могу да се сврстају и она која описују како се добија иницијални распоред свих пријављених такмичара по мечевима, такозвани жреб.

Сваки ЕР дијаграм мора да има ентитете и везе. Понекад имамо ситуације које није једноставно моделовати па су нам потребне још неке посебне структуре. Објаснићемо под и надтипове, лукове, хијерархију, рекурзију, као и генерички модел које по потреби можемо да користимо приликом моделовања.

## 3.4. Подтипови и лукови

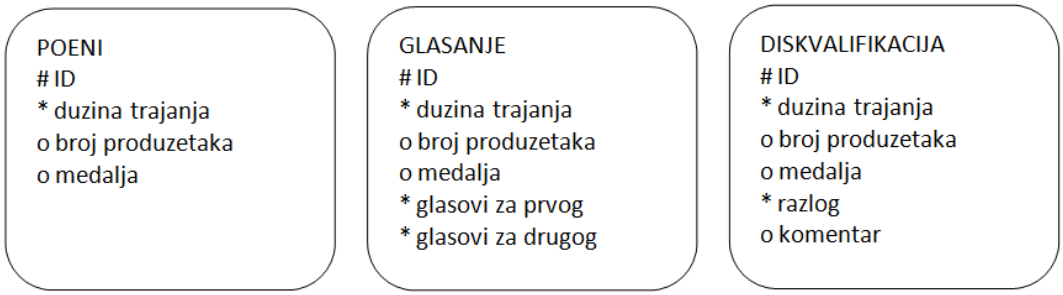
Сваки меч има крајњи исход, односно резултат. Резултатом меча се дефинише победник, време колико је трајао меч (да ли је било продужетака) и да ли је последица меча додела неке од медаља победнику. Меч може да се заврши на три начина. Уколико је један од такмичара освојио више поена од другог, у току регуларног дела меча или током продужетака, проглашава се победником. Уколико је резултат нерешен и након завршетка продужетака, судије се изјашњавају гласањем. Меч може да се заврши и пре истека времена уколико је један од такмичара дисквалификован. До дисквалификације може да дође уколико се такмичар не појави или направи неки озбиљан прекршај. Имамо доста података који описују резултат меча, тако да ћемо на дијаграму имати ентитет *Rezultat*, за који уводимо вештачки примарни идентификатор у недостатку природног. За сваки од могућих исхода је потребно чувати и неке специфичне податке, као што је на пример разлог дисквалификације или број гласова судија након гласања. Различити исходи меча су подтипови ентитета *Rezultat*. Победник меча је такмичар који постоји као ентитет. Из тог разлога ћемо победника меча приказати везом ка ентитету *Takmicar*, уместо да га чувамо као атрибут. Ентитет *Rezultat* је повезан и са ентитетом *Mec* јер је сваки резултат исход неког меча. Ентитет *Rezultat* са свим подтиповима је приказан на следећој слици.




Слика 19. Ентитет *Rezultat* са подтиповима

Подтип *Poeni* нема посебне атрибуте јер поени сами за себе имају податке који их дефинишу (време када је дошло до поена, тип поена, за коју технику, да ли је поен резултат казне...) па ће постојати ентитет *Poeni* уз помоћ којег може да се дође и до резултата појединачним пребројавањем освојених поена у једном мечу.

Сваки од подтипова има све атрибуте надтипа, као и своје посебне атрибуте. Тако да све атрибуте које имају подтипови можемо да видимо на следећој слици.



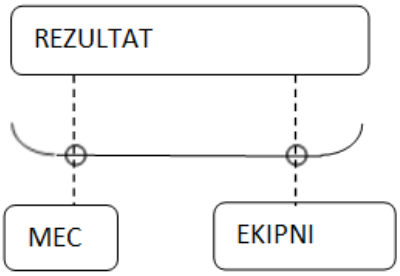

Слика 20. Ентитети *Poeni*, *Glasanje* и *Diskvalifikacija*

Слично смо могли, уместо да цртамо један надтип са свим подтиповима, да имамо на ЕР дијаграму ова три различита ентитета. Ово би искомпликовало сам дијаграм јер сва три ентитета учествују у везама са ентитетима *Mec* и *Takmicar*, па би уместо две везе које смо имали на претходном цртежу са надтипом *Rezultat*, имали шест веза што доводи до компликованијег дијаграма. Опште правило је да користимо под и надтипове у ситуацијама кад неколико ентитета има пуно заједничких атрибута (толико су слични и нема потребе да исте атрибуте понављамо на више места) или учествују у много истих веза.

Свака инстанца надтипа, у овом слулају ентитета *Rezultat*, мора да буде инстанца тачно једног подтипа. Тако је инстанца надтипа *Rezultat* или *Poeni*, или *Glasanje* или *Diskvalifikacija*. Не може да постоји инстанца ентитета *Rezultat* која није *Poeni*, *Glasanje* или *Diskvalifikacija*. Поред тога, подтипови морају да буду међусобно искључиви. Уколико је меч завршен гласањем, онда се није

завршио изгласавањем судија или дисквалификацијом, тј. меч не може да се заврши и на поене и дисквалификацијом у исто време. Подтипови морају и да покрију све могуће различите типове надтипа. Уколико би изоставили неки од ових подтипова, на пример подтип *Diskvalifikacija*, не би могли да забележимо исправно резултат меча који се завршио дисквалификовањем једног од такмичара. Да би модел био флексибилнији и отворенији за појаву и неког другог подтипа, који можда није могао да се уочи у почетној фази пројектовања, често се додаје подтип *Ostalo*. Тај подтип није обавезан и није ни пожељан у ситуацијама када имамо тачно прецизиран и фиксиран број подтипова.

На следећој слици можемо да видимо лук. Атрибути су изостављни да би цртеж био једноставнији.




Слика 21. *Лук*

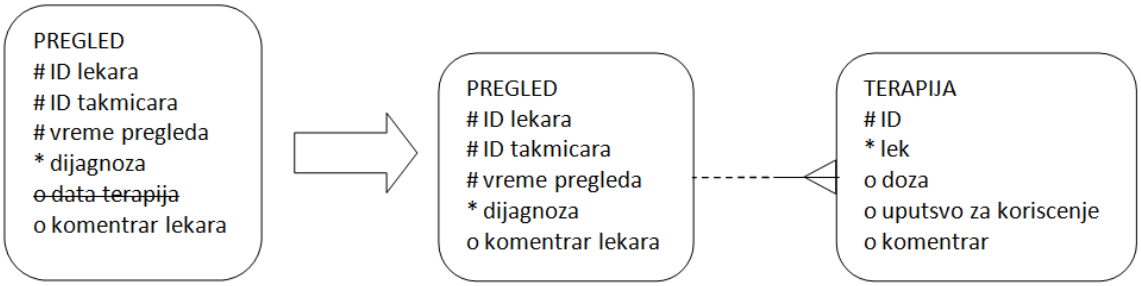
Луком представљамо међусобно искључиве везе. На такмичењима имамо екипне и појединачне мечеве. Екипни се састоје из више појединачних. Сваки меч, и појединачни и екипни, има резултат. Један резултат је исход или појединачног меча или екипног меча. Једна инстанца ентитета *Rezultat* је у вези или са инстанцом ентитета *Mec* или са инстанцом ентитета *Ekipni mec*, али никако са оба у исто време. То може да се представи луком.

## 3.5. Нормализација

Постоје неформална правила која гарантују конзистентност података у бази. Морамо да чувамо све податке који су нам неопходни за пословање. База није комплетна и нећемо моћи да добијемо потребне информације уколико не чувамо све битне податке. Подаци морају да се чувају на тачно једном месту у бази података и то на месту које је најбоље место за њих. На пример, сви подаци о клубу (ПИБ број, назив, адреса, телефон...) ће се чувати у ентитету *Клуб* и неће се појављивати на другим местима у бази. Претпоставимо да ситуација није таква и да број телефона клуба чувамо на неколико различитих места у бази података. Уколико се број телефона промени и ми га изменимо у бази података, може да се деси да га изменимо на једном или два места, али да случајно стари број остане на трећем. Након неког времена ћемо приметити да, ако поставимо захтев да се испише број телефона, ће се дешавати да нам база врати различите вредности и нећемо знати која је тачна. Још једно од неформалних правила којег треба да се придржавамо је да не чувамо податке који могу да се изведу или израчунају из других. Један од разлога је бесмислено заузимање више меморије него што је потребно. Други разлог је исти као код броја телефона, јер уколико почетне вредности изменимо, може да се деси да се нова изведена вредност не израчуна благовремено и да се тиме наруши конзистентност података.

Постоје и фомална правила која се зову правила нормализације, према [1], и која гарантују конзистентност података у бази. По првом правилу нормалне форме (ознака 1NF), не смеју да постоје атрибути који за једну инстанцу имају више вредности.

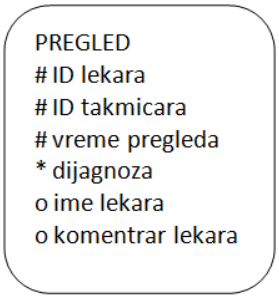
Током једног прегледа, може да се да и неколико различитих терапија, односно лекова. На пример, може да се обави дезинфекција ране, превијање и да се дају лекови против болова. Свака од тих терапија би требала да се забележи ради евиденције о употребљеним медицинским средствима, као и да би се формирала комплетна документација о лечењу. У том случају атрибут *Terapija* би имао више вредности за једну инстанцу ентитета *Pregled*. Тиме је нарушено правило прве нормалне форме и модел мора да се поправи. Атрибут који може да има више вредности се издваја у посебан ентитет и повезује са ентитетом у којем се налазио везом типа 1:M.




Слика 22. Нарушење правила Прве нормалне форме и решење проблема

Друго правило нормалне форме (ознака 2NF) каже да сви атрибути који нису део примарног јединсвеног идентификатора морају да буду у пуној зависности од целог примарног јединственог идентификатора. Суштина овог правила је у томе да сви атрибути треба да буду на правом месту, односно баш унутар ентитета на који се односе.

До нарушења правила друге нормалне форме би могло да дође у следећој ситуацији. Замислимо да ентитет *Pregled* садржи, поред атрибута које смо већ набројали, и име лекара.

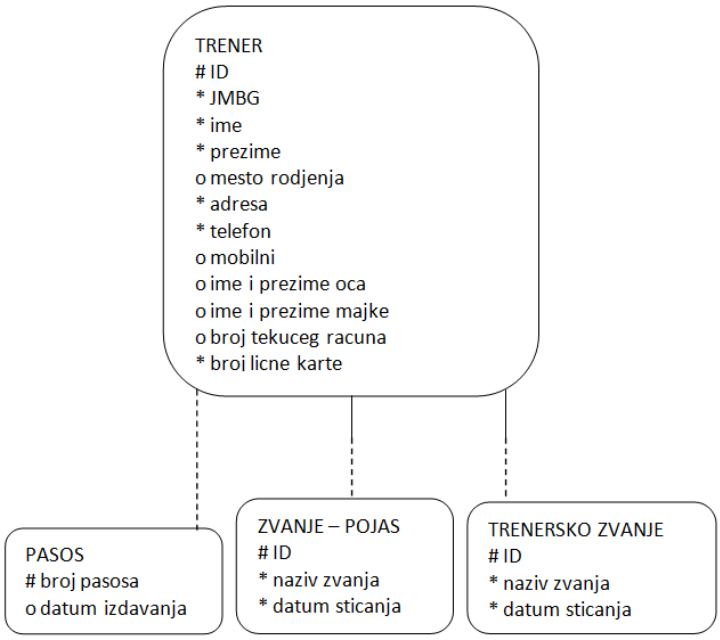



Слика 23. Нарушење правила Друге нормалне форме

Атрибути, који нису део примарног јединственог идентификатора, су: *dijagnoza, ime lekara* и *komentar lekara*. Дијагноза и коментар лекара се односе на конкретну инстанцу ентитета *Pregled* и у пуној су зависности од целог примарног јединственог идентификатора. Дијагнозу и коментар је одређени лекар дао неком такмичару у назначено време прегледа. За разлику од ова два атрибута, атрибут *ime lekara* је везан за шифру лекара, и потпуно је независан од такмичара (чија је шифра део примарног јединственог идентификатора) и времена прегледа. Лекар се зове, на пример, Милош, и тако се зове и док прегледа било ког такмичара у било које доба дана. Када је нарушено друго правило нормалне форме, потребно је спорни атрибут избацити из ентитета и проверити да ли се налази у ентитету где има смисла да се налази. У овом случају, атрибут *ime lekara* треба да се налази у ентитету *Lekar*. Можемо да приметимо да, уколико смо се придржавали неформалних правила, атрибут *ime lekara* се никад не би ни нашао у ентитету

*Pregled*. Међутим, дијаграми су често јако велики и може да се деси да се поткраде оваква грешка. Зато је битно да се дијаграм преконтролише и да се провери да ли испуњава све услове нормализације, а уколико то није случај да се грешке исправе.

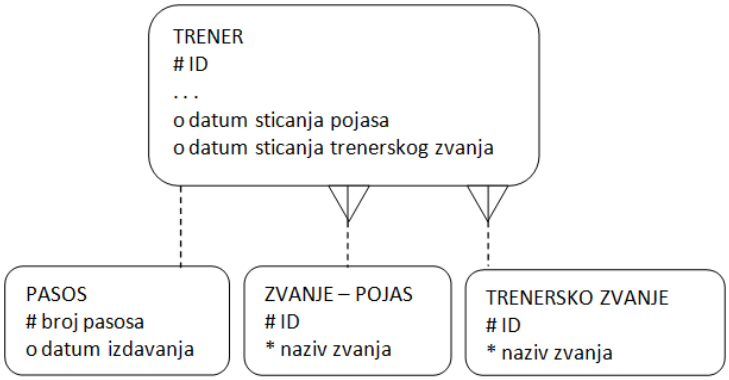
Треће правило нормалне форме (ознака 3NF) говори о томе да не сме да постоји узајамна зависност међу атрибутима који нису део примарног јединственог идентификатора. Суштина овог правила је у томе да, ако постоји група атрибута унутар ентитета која представља повезану целину за себе, треба да се издвоје у засебан ентитет. Погледајмо поново ентитет *Trener*. Атрибути *broj pasosa* и *datum izdavanja* чине целину која се односи на пасош. Уколико тренер извади нови пасош и добије нови број, промениће се и датум издавања. Слична ситуација је и са атрибутима: *zvanje – pojas* и *datum sticanja zvanja*, као и са атибутима: *trenersko zvanje* и *datum sticanja trenerskog zvanja*. Све поменуте атрибуте је потребно уклонити из ентитета *Trener* и формирати нове ентитете, као што је приказано следећом сликом.




**Слика 24. Решење проблема који је настао јер је нарушено павило Треће нормалне форме**

Иако то овде није случај, веза која се добије након што се уклони проблем који је кршио треће правило нормалне форме, је често типа 1:M. У овом примеру би везе ка ентитетима *Zvanje* и *Trenersko zvanje* имале другачију кардиналност, тј. биле би типа 1:M (један тренер:више звања), уколико бисмо имали потребу да чувамо сва звања која је један тренер током живота стицао. Уколико би се правила база података за неки конкретан клуб, то и јесу битни подаци које бисмо чували. У случају као што је овај, за потребе турнира, није потребно чувати све те податке, већ само имати податак о последњем, највећем, стеченом звању.

Може да се уочи да велики број тренера има исто звање које додељује одговарајући спортски савез. За сваког од њих се међутим разликује датум када су га стекли. У том случају може да се као ентитет *Zvanje* уочи опште звање које се додељује тренерима, а податак који говори о томе када је стечено то највише звање треба да се смести у ентитет *Trener*. Иста ситуација која је претходно приказана, може да се моделира и на други начин, приказан на следећој слици. Приликом оваквог моделовања може да се уочи да исто звање може да има више тренера па је веза типа 1:M (једно звање:више тренера).

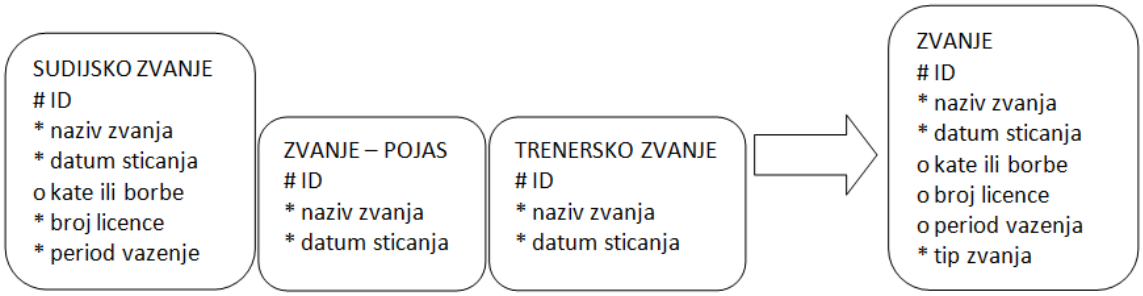



Слика 25. Везе између ентитета *Trener* и ентитета *Pasos*, *Zvanje – pojas* и *Trenersko zvanje*

Често у фази моделовања можемо да уочимо и неколико различитих решења проблема везаног за исту ситуацију. Потребно је добро размотрити сва решења, њихове предности и мане, и одлучити се за једно. За модел турнирског такмичења је изабрано прво од два управо описана решења.

## 3.6. Генерализација

У претходном примеру може да се уочи да су ентитети *Zvanje – pojas* и *Trenersko zvanje* јако слични. Осим тренера, на турнирима су присутне и судије које имају своја звања која могу да се представе ентитетом *Sudijsko zvanje*. Ова три ентитета су јако слична и са готово идентичним атрибутима. У овој ситуацији дијаграм може да се упрости поступком који се назива генерализација. Могуће је креирати један ентитет *Zvanje* чије ће инстанце бити: појас, тренерско звање или судијско звање. Приликом генерализације је потребно додати још један атрибут који ће означавати којег је типа конкретна инстанца. Уколико неки од почетних ентитета имају атрибуте које други ентитети немају, у генеричком моделу ти атрибути морају да постану опциони, без обзира на њихову почетну опционалност, а оригинална опционалност треба да се документује. Док тренери само прате такмичаре на турниру, судије имају много већу улогу јер суде у мечевима. Потребно је да имају, поред стечених звања и одговарајуће лиценце за категорије за коју суде. Због тога се ентитет *Sudijsko zvanje* мало разликује од друга два ентитета и потребно је обратити пажњу на његове посебне атрибуте: *kate ili borbe*, *broj licence* и *period vazenja*.



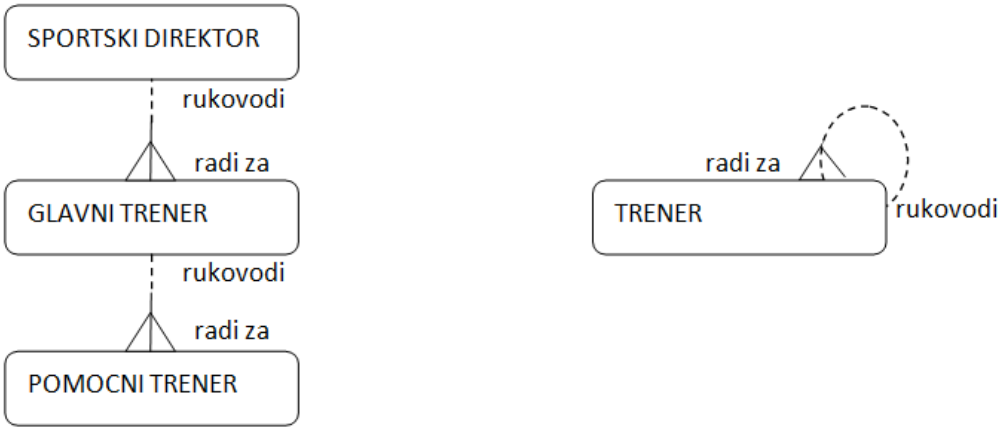

Слика 26. Генерализација

Обавезан атрибут *tip zvanja* у генеричком ентитету може да има једну од три вредности: појас, судијско звање или тренерско звање. Уколико је инстанца ентитета *Zvanje* типа појас, не постоји вредност атрибута *broj licence* и *period vazenja*, док ти атрибути морају да имају вредност код свих инстанци које су судијско звање.

Постоје ситуације и када генерализација нема смисла. Ентитети *Sudija*, *Trener*, *Takmicar* и *Lekar* су јако слични и имају велики број истих атрибута (име, презиме, ЈМБГ, адреса, телефон...), али учествују у потпуно различитим везама тако да је боље да се оставе као појединачни ентитети. Генерализација у њиховом случају би довела до непотребног компликовања и непрегледности дијаграма.

## 3.7. Хијерархија и рекурзија

У једном клубу често ради више тренера који имају различите улоге али и различите позиције у оквиру хијерархије клуба. Често имамо једног тренера који је спортски директор клуба. Затим можемо да имамо више главних тренера који су задужени за различите групе вежбача. На дну хијерархије се налазе помоћни тренери који асистирају главним тренерима током тренинга. Хијерархијска организација је честа, како у спорту, тако и у разним компанијама. Приликом моделовања ове ситуације може да се употреби или хијерархијски модел или рекурзивна веза. Иако везе у хијерархијском моделу могу да буду и опционе и обавезне (помоћни тренер мора да ради за тачно једног главног тренера), у рекурзивном моделу веза мора да буде опциона целом дужином због оних који се налазе на дну или на врху хијерархије. Рекурзивна веза иде од ентитета ка истом ентитету. У овом случају приказује однос тренера који руководи другим тренерима, односно за сваког тренера указује на његовог надређеног. На левој страни следеће слике је приказано хијерархијско, а на десној рекурзивно решење истог проблема.




Слика 27. Хијерархијска структура и рекурзија

Организаторима турнира није битно да познају тачни хијерархијски однос међу тренерима који су дошли у пратњи такмичара, тако да ћемо са дијаграма изоставити ову рекурзивну везу.

## 4. Модел турнирског такмичења

Пре објављивања позива на турнир, организатори прецизирају категорије у које ће бити подељени такмичари. Категорија је важан појам и зато је на моделу представљена ентитетом. За званична државна и међународна првенства се поштују препоруке одговарајуће светске федерација. За мање турнире је могуће да се категорије мало прилагоде према очекиваном броју такмичара. Ако су две категорије које су препоручене од стране федерације јако сличне, а очекује се да ће се у обе пријавити мали број такмичара, организатори турнира могу да их обједине. Такмичари су у категорије подељени првенствено по години рођења. Некад је категорија таква да су у њој сви такмичари рођени исте године, а некад имамо распон од две до три године. За сениоре, најстарије такмичаре, се прецизира само годиште најмлађих такмичара, док је горња граница недефинисана. За најмлађе такмичаре на турниру се прецизира годиште најстаријих такмичара. Како најстарији имају прецизирану само доњу, а најмлађи само горњу границу, и горња и доња граница морају да буду опционе на ЕР дијаграму, али је потребно да документујемо процедурално правило да бар један од атрибута *min godiste* и *maks godiste* мора да има вредност. Наравно да ће у највећем броју случајева оба имати уписану вредност.

Свака категорија има назив. Назив може да буде прецизиран од стане федерације, на пример: пионири, наде, кадети, јуниори, сениори... Може да буде и описан, састављен од годишта и других карактеристика, на пример: „2002-2003 кате појединачно“. Организатори турнира могу да уведу и своје ознаке, на пример: „1. група“. Све категорије су углавном подељене у две велике групе (кате и борбе) и зато се уводи *tip kategorije* као посебан ентитет који прецизира којој групи припада која категорија. На следећој слици следи приказ ката категорија једног турнира преузете са [6].

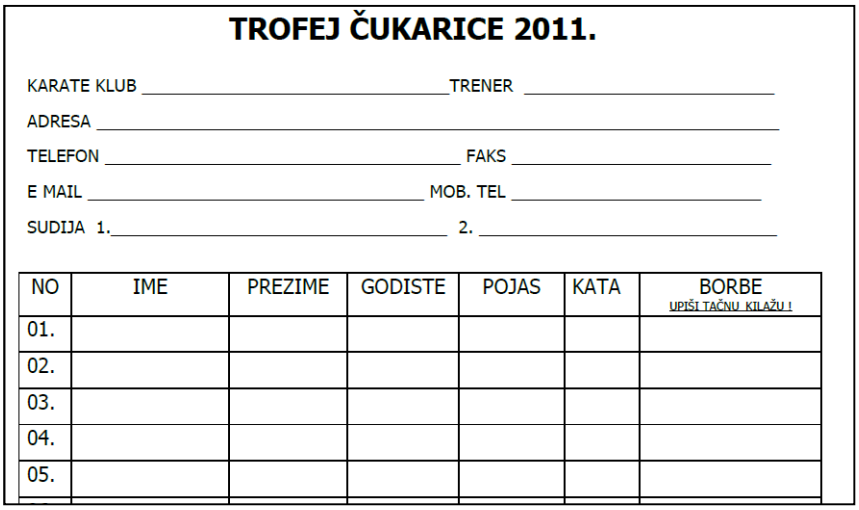
| GODIŠTE | POJEDINAČNO | | | EKIPNO |
|---|---|---|---|---|
| | POJAS | | | |
| 2004 i mlađi | ŽUTI | ORANŽ | | 1. GRUPA |
| 2003 | ŽUTI | ORANŽ | APS (svi pojasevi) | |
| 2002 | ŽUTI | ORANŽ | APS (svi pojasevi) | 2. GRUPA |
| 2001 | ŽUTI-ORANŽ | ZELENI-PLAVI | APS (svi pojasevi) | |
| 2000 | ŽUTI-ORANŽ | ZELENI-PLAVI | APS (svi pojasevi) | 3. GRUPA |
| 1999 | ŽUTI-ORANŽ | ZELENI-PLAVI | APS (svi pojasevi) | |
| 1998 | ŽUTI-ORANŽ | ZELENI-PLAVI | APS (svi pojasevi) | 4. GRUPA |
| 1997 | ZELENI-PLAVI | | APS (svi pojasevi) | |
| 1996 | ZELENI-PLAVI | | APS (svi pojasevi) | 5. GRUPA |
| 1995 | APS (svi pojasevi) | | | |

Слика 28. Списак ката категорија на турниру „Трофеј Чукарице 2011“

За категорије ката су често битни појасеви. Два такмичара истог годишта који имају жути и црни појас се неће такмичити у истој категорији. За борбе је битна тежина, тако да се такмичар који је тежак, на пример, 40kg неће бити у неправедној ситуацији да се бори са противником који је исто годиште али има око 60kg. Свака категорија може да има и прецизиран списак посебних правила. Ова правила могу да се разликују од турнира до турнира. Правила подразумевају, на пример, списак ката које су дозвољене у одређеним колима турнира, као и то да ли су дозвољени мешани тимови који се састоје од дечака и девојчица у млађим узрасним категоријама.

На следећој слици је приказан формулар за пријаву на један турнир који не спада у званична првенства и који се може преузети са [6]. Ту је приказан неки минимум података о клубовима, такмичарима, тренерима и судијама који се захтева приликом припреме турнира. Сви клубови попуне и пошаљу ову пријаву да би организатори турнира знали оквиран број учесника. На сам

дан турнира је организаторима потребно још података [4]. Сви такмичари са собом на турнир донесу такмичарске књижице које садрже основне личне податке. Поред тога, за судије је, на пример, битно да се зна и број текућег рачуна на који ће организатори уплатити хонорар за суђење.



**TROFEJ ČUKARICE 2011.**

KARATE KLUB _____ TRENER _____

ADRESA _____

TELEFON _____ FAKS _____

E MAIL _____ MOB. TEL _____

SUDIJA 1._____ 2. _____

| NO | IME | PREZIME | GODISTE | POJAS | KATA | BORBE UPIŠI TAČNU KILAŽU ! |
|---|---|---|---|---|---|---|
| 01. | | | | | | |
| 02. | | | | | | |
| 03. | | | | | | |
| 04. | | | | | | |
| 05. | | | | | | |

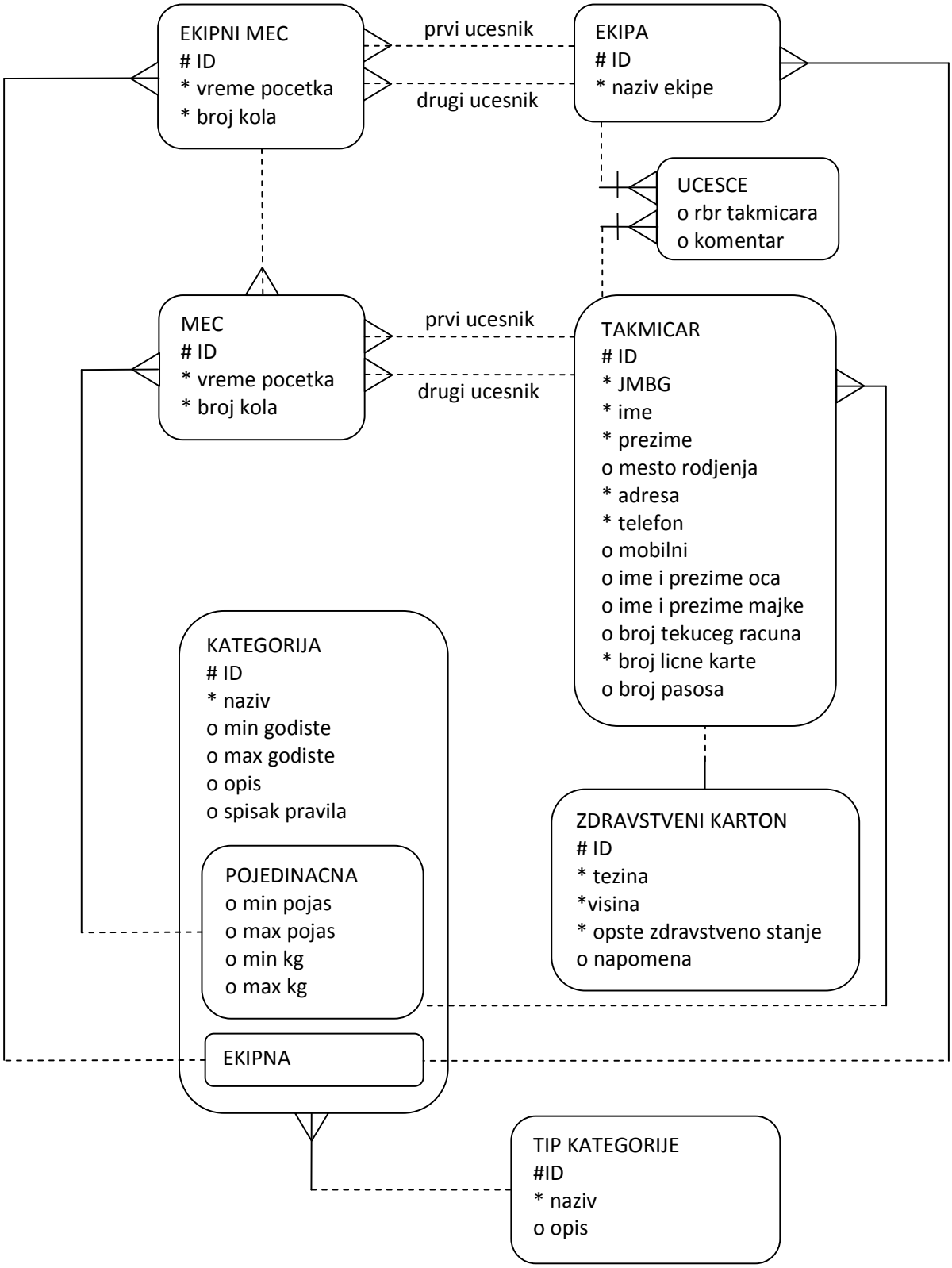
Слика 29. Формулар за пријаву на турнир „Трофеј Чукарице 2011“

На турниру се одржавају мечеви. У сваком мечу учествују два такмичара и важно је да се зна који је носио плави, а који црвени појас (први и други учесник). Екипни сусрет две екипе подразумева да се одржи одређени број појединачних мечева. Сваки екипни меч може да се састоји од једног или више мечева. Сваки меч може да буде део једног и само једног екипног меча. Процедурално правило које је важно да се овде забележи је да, уколико се сређу женске екипе тај број је три, а код мушких је пет. Јасна је опционалност на страни ентитета *Mec* јер сваки меч може да буде и у појединачној конкуренцији и у том случају није део екипног сусрета. Међутим, опционалност је и на страни ентитета *Ekipni mec* јер и уколико дође до екипног меча, може да се деси да се не одржи ни један од мечева уколико је дошло до дисквалификације једног од тимова.

Иако јако ретко, може да се деси да један такмичар учествује у екипним такмичењима и у борбама и у катама, па може да буде члан више од једне екипе. Јасно је да једну екипу чини више такмичара. Веза између ентитета *Takmicar* и *Ekipa* је била типа M:M. Уместо ње је формиран нови ентитет *Ucesce*.

Сви такмичари су обавезни да имају здравствени картон у установама задуженим за спортисте и да увек са собом носе извештај лекара који је уписан у такмичарску књижицу са осталим личним подацима. Додатно се на такмичењу преконтролише тежина такмиачара уколико се такмичи у борбама.

Пратећа процедурална правила:
Такмичар може да учествује само у мечевима у оним категоријама за које је он пријављен.
Сви мечеви екипног меча морају да имају исти број кола као и екипни меч.
Тежина такмичара мора да буде већа или једнака минималној и мања или једнака максималној тежини у категорији у којој се такмичи.
Такмичар не може да учествује у мечу са већим бројем кола уколико није победио у свим мечевима које је водио у претходним колима.


EKIPNI MEC
# ID
* vreme pocetka
* broj kola
prvi ucesnik
drugi ucesnik
EKIPA
# ID
* naziv ekipe
UCESCE
o rbr takmicara
o komentar
MEC
# ID
* vreme pocetka
* broj kola
prvi ucesnik
drugi ucesnik
TAKMICAR
# ID
* JMBG
* ime
* prezime
o mesto rodjenja
* adresa
* telefon
o mobilni
o ime i prezime oca
o ime i prezime majke
o broj tekuceg racuna
* broj licne karte
o broj pasosa
KATEGORIJA
# ID
* naziv
o min godiste
o max godiste
o opis
o spisak pravila
POJEDINACNA
o min pojas
o max pojas
o min kg
o max kg
EKIPNA
ZDRAVSTVENI KARTON
# ID
* tezina
*visina
* opste zdravstveno stanje
o napomena
TIP KATEGORIJE
#ID
* naziv
o opis

Најважније дешавање на турниру представљају мечеви, па на дијаграму мора да постоји ентитет *Mec*. Свако коло се састоји од више мечева и победници из тих мечева прелазе у наредно коло. За меч је важно да се забележи тренутак почетка. Остали подаци који су саставни део сваког меча су толико битни да су на дијаграму приказани засебним ентитетима и повезани са ентитетом *Mec*. Такмичари у мечу остварују поене и, на првом месту је исход меча одређен поенима који се освоје. Током једног меча може да се додели више поена, једном или обојици такмичара. Исти такмичар током једног меча може да добије више поена. Веза „освојити поен" између ентитета *Mec* и *Takmicar* је била типа M:M. Уместо ње се уводи ентитет *Poen*. У тренутку постизања поена меч се зауставља и бележи се време, тренутак у мечу када је дошло до поена. Судија уочава поентирајућу технику коју је такмичар извео (назив ударца руком или ногом), као и ударну површину (тело, глава или леђа противника). Приликом чишћења, обарања противника на под, може да се деси да је уследио или изостао ударац, па је потребан и додатан коментар о укупном поену. Тип поена увек има назив (на пример: ипон, нихон, самбон) као и број бодова које носи (редом: један, два или три). За одређене акције могу да се доделе и казне такмичарима. Неке казне резултују поенима за другог такмичара.

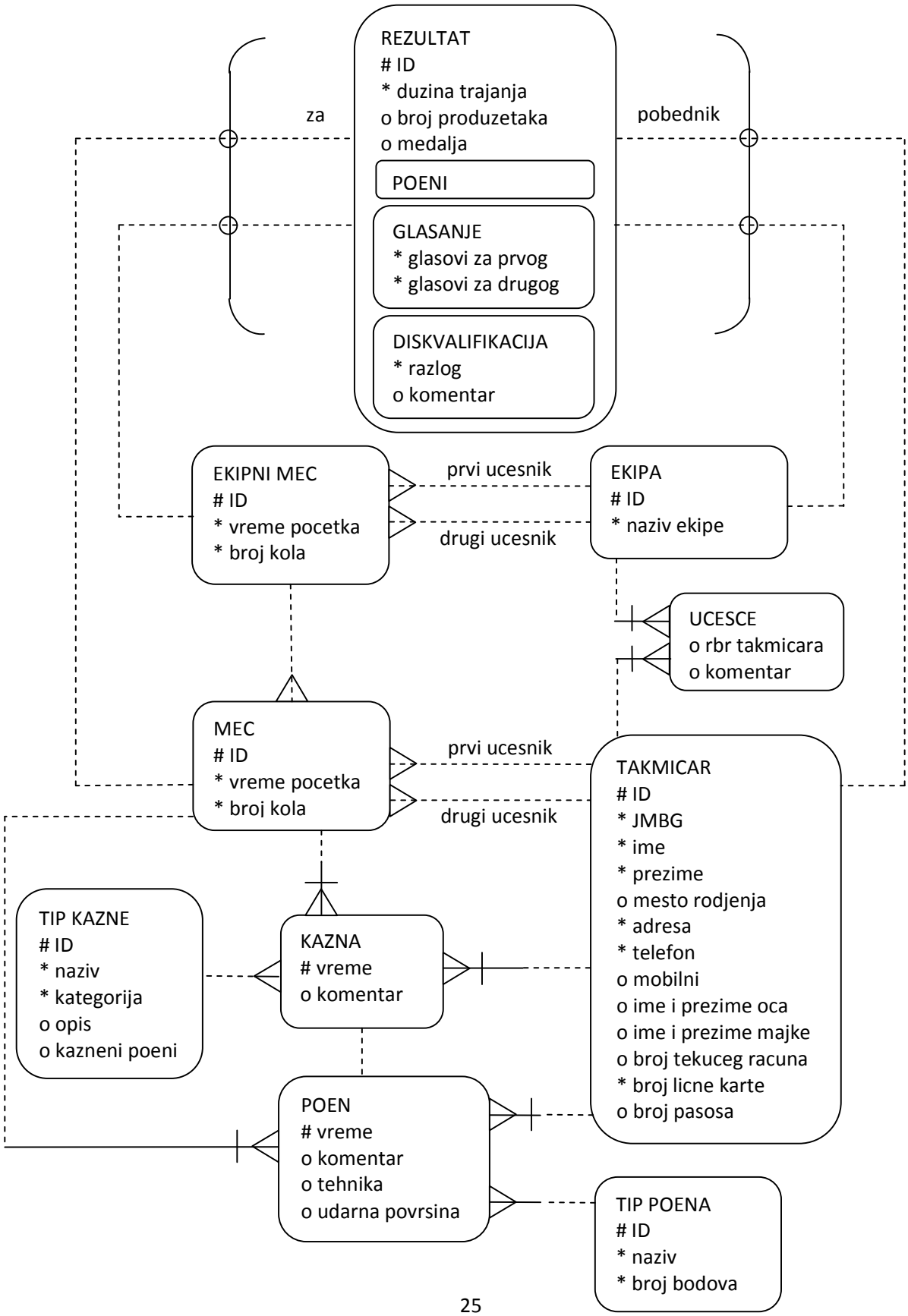
Сваки меч се одвија на тачно једном борилишту. Организатори турнира распореде на свако борилиште неколико записничара. Они на том борилишту проведу цео дан бележећи податке о сваком одржаном мечу на том борилишту. Записничари не мењају борилишта током дана, за разлику од судија. Судије се у групама поделе по борилиштима. Организатори турнира распореде и различите категорије по борилиштима. Борилишта су нумерисана. На пример, на борилишту број 3 ће се такмичити дечаци 2001. годиште, оранж и зелени појасеви, у катама. Током дана се направи и неколико пауза, углавном када се сви мечеви једне категорије заврше. Након паузе, група судија може да буде преорганизована по другим борилиштима. Судије могу међусобно да се договарају о томе ко ће у ком мечу, или групи мечева, да буде главни судија. У сваком мечу је један судија главни, а остали су помоћни. То је на дијаграму приказано везом тима 1:1 између ентитета *Mec* и ентитета *Sudija*. Ентитет *Sudjenje* је настао као разрешење везе тима M:M која је описивала однос ентитета *Sudija* и *Mec*. На сваком мечу може да суди један или више судија. Сваки судија може да суди на једном или више мечева. У случају повреде, меч се зауставља и такмичар одлази на преглед код лекара или лекар долази на борилиште да прегледа такмичара. Бележе се важни подаци о прегледу, као и о датим терапијама.

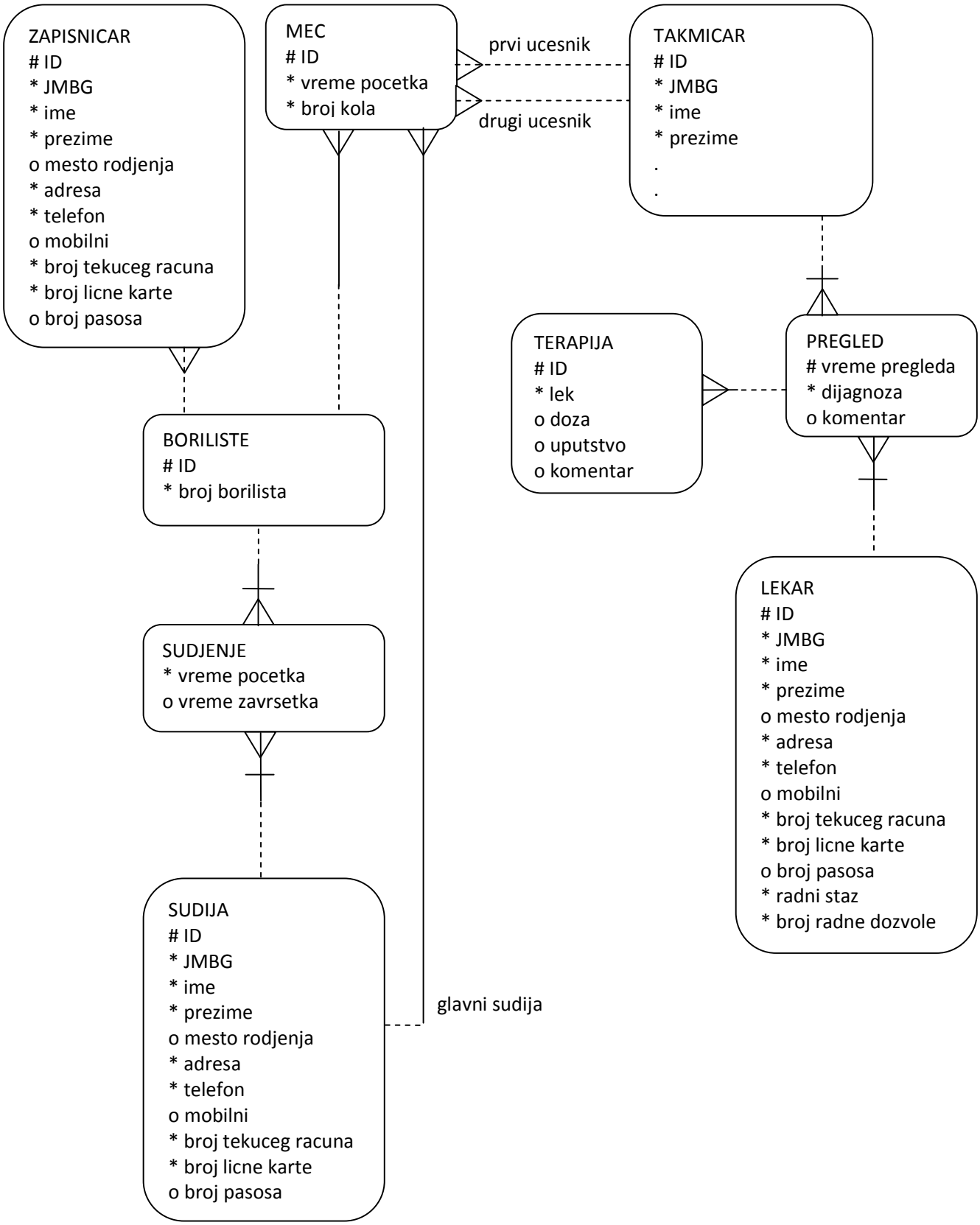
Клубови пријављују такмичаре на турнир. Такмичари долазе у пратњи тренера. Клубу се издаје рачун. На рачуну се наводе све ставке, списак котизација за све такмичаре и екипе. За различите категорије може да буде различита котизација. То је представљено ентитетом *Stavka za naplatu*. Иста ставка може да се нађе на више рачуна јер више клубова пријављује такмичаре у истој категорији. Исти клуб може да плати више котизација (за појединачне такмичаре, за екипе...). Веза типа M:M се замењује новим ентитетом.

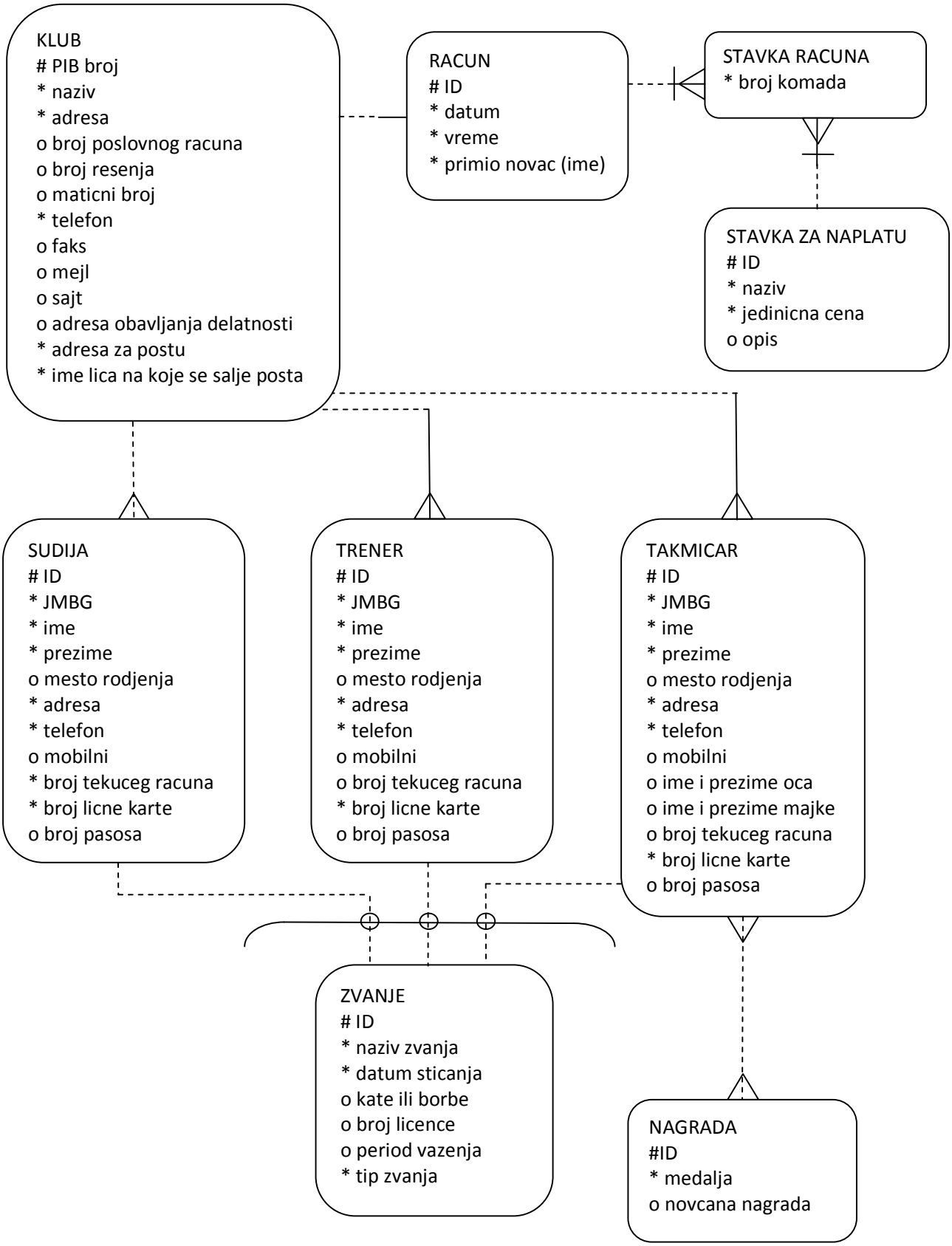
Тренери и такмичари морају да буду чланови неког клуба који их је и довео на турнир. Из тог разлога је веза од ентитета *Такмичар* и *Тренер* ка ентитету *Клуб* обавезна. Судија може да буде независан, тј. да не припада ниједном клубу, па је код њега веза опциона.

Пратећа процедурална правила:
Тип поена мора да одговара комбинацији техника и ударна површина прописаним правилником одговарајуће светске федерације.
У једном тренутку на једном борилишту мора да буде тачно пет судија.
Казна која носи мањи број поена за другог такмичара може да се додели само уколико није већ изречена казна која носи већи број поена.

Описана правила организовања турнира могу да се прикажу следећим дијаграмима.


REZULTAT
# ID
* duzina trajanja
o broj produzetaka
o medalja
POENI
GLASANJE
* glasovi za prvog
* glasovi za drugog
DISKVALIFIKACIJA
* razlog
o komentar
za
pobednik
EKIPNI MEC
# ID
* vreme pocetka
* broj kola
prvi ucesnik
drugi ucesnik
EKIPA
# ID
* naziv ekipe
UCESCE
o rbr takmicara
o komentar
MEC
# ID
* vreme pocetka
* broj kola
prvi ucesnik
drugi ucesnik
TAKMICAR
# ID
* JMBG
* ime
* prezime
o mesto rodjenja
* adresa
* telefon
o mobilni
o ime i prezime oca
o ime i prezime majke
o broj tekuceg racuna
* broj licne karte
o broj pasosa
TIP KAZNE
# ID
* naziv
* kategorija
o opis
o kazneni poeni
KAZNA
# vreme
o komentar
POEN
# vreme
o komentar
o tehnika
o udarna povrsina
TIP POENA
# ID
* naziv
* broj bodova


ZAPISNICAR
# ID
* JMBG
* ime
* prezime
o mesto rodjenja
* adresa
* telefon
o mobilni
* broj tekuceg racuna
* broj licne karte
o broj pasosa
MEC
# ID
* vreme pocetka
* broj kola
prvi ucesnik
drugi ucesnik
TAKMICAR
# ID
* JMBG
* ime
* prezime
.
.
TERAPIJA
# ID
* lek
o doza
o uputstvo
o komentar
PREGLED
# vreme pregleda
* dijagnoza
o komentar
BORILISTE
# ID
* broj borilista
SUDJENJE
* vreme pocetka
o vreme zavrsetka
LEKAR
# ID
* JMBG
* ime
* prezime
o mesto rodjenja
* adresa
* telefon
o mobilni
* broj tekuceg racuna
* broj licne karte
o broj pasosa
* radni staz
* broj radne dozvole
SUDIJA
# ID
* JMBG
* ime
* prezime
o mesto rodjenja
* adresa
* telefon
o mobilni
* broj tekuceg racuna
* broj licne karte
o broj pasosa
glavni sudija


KLUB
# PIB broj
* naziv
* adresa
o broj poslovnog racuna
o broj resenja
o maticni broj
* telefon
o faks
o mejl
o sajt
o adresa obavljanja delatnosti
* adresa za postu
* ime lica na koje se salje posta
RACUN
# ID
* datum
* vreme
* primio novac (ime)
STAVKA RACUNA
* broj komada
STAVKA ZA NAPLATU
# ID
* naziv
* jedinicna cena
o opis
SUDIJA
# ID
* JMBG
* ime
* prezime
o mesto rodjenja
* adresa
* telefon
o mobilni
* broj tekuceg racuna
* broj licne karte
o broj pasosa
TRENER
# ID
* JMBG
* ime
* prezime
o mesto rodjenja
* adresa
* telefon
o mobilni
o broj tekuceg racuna
* broj licne karte
o broj pasosa
TAKMICAR
# ID
* JMBG
* ime
* prezime
o mesto rodjenja
* adresa
* telefon
o mobilni
o ime i prezime oca
o ime i prezime majke
o broj tekuceg racuna
* broj licne karte
o broj pasosa
ZVANJE
# ID
* naziv zvanja
* datum sticanja
o kate ili borbe
o broj licence
o period vazenja
* tip zvanja
NAGRADA
#ID
* medalja
o novcana nagrada

# 5. Превођење логичког модела у физичку базу података

Реалациону базу података, физички модел животног процеса, корисници виде као колекцију дводимензионих табела. Сви подаци се чувају у пољима тих табела. Пре физичког креирања базе података, потребно је да се логички модел, тј. дијаграм ентитета и веза, прилагоди релационом моделу. Тај процес се назива и превођење логичког у физички модел([1]), или другачије мапирање (енгл. Mapping).

Долази до промене у терминологији. Ентитети са логичког модела ће постати табеле у бази података. Атрибути постају колоне. Инстанце ентитета постају редови у табели. Примарни јединствени идентификатор постаје примарни кључ. Кандидати за примерни јединствени идентификатор постају ограничења јединствености. Везе постају страни кључеви.

Постоје и правила која гарантују интенгритет података у бази (енгл. Data-integrity rules) која се другачије називају и ограничења (енгл. Constraints).

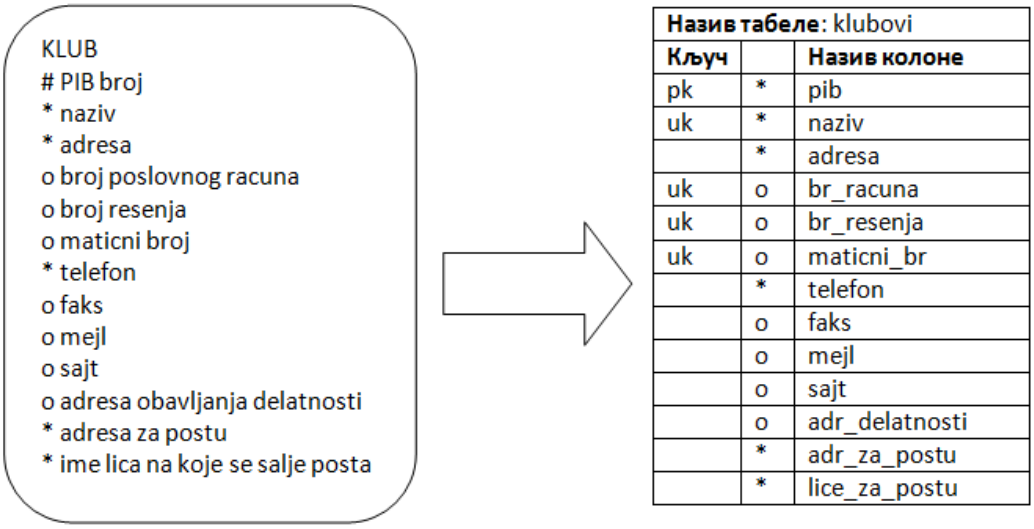
Интегритет ентитета чине правила која се односе на табеле у бази података. Табела мора да има име које је у складу са конвенцијама именовања објеката у бази. Примарни кључ табеле мора да буде јединствен и ниједан део примарног кључа не сме да буде недефинисан. За недефинисан податак у базама података се уводи константа *null*.

Имена табела су углавом именице у множини које одговарају именицама у једнини којима су били названи ентитети. Тако ће, на пример, ентитет *Trener* постати табела *Treneri*, а ентитет *Klub* постати табела *Klubovi*.

Правила за именовање табела се односе и на именовање колона. Име мора да почне словом. Највећа дужина је 30 карактера који су слова или цифре. Не сме да садржи размаке или неке друге знаке, као што су интерпукцијски знаци. Знаци: $, # и _ су дозвољени. Име табеле у оквиру једне шеме, као и име колоне у оквиру једне табеле, мора да буде јединствено. Шема представља скуп свих објеката које креира један корисник базе. У различитим шемама могу да постоје табеле са истим именом, као што и у различитим табелама могу да постоје колоне са истим именом. Тако је име колоне *id* често у табелама и увек се односи на јединствени идентификациони број у табели у којој се налази. На пример, колона *id* у табели *Takmicari* се односи на јединствени идентификатор такмичара, док се колона *id* у табели *Treneri* односи на идентификатор тренера.

За сваки ентитет са логичког модела се креира једна табела са три колоне. Сваки ред те табеле представља податак о једном атрибуту, тј. колони у бази. У првој колони се наведе назив кључа за ту колону. У другој се прецизира опционалност: звездица (*) за колоне у којима мора да буде уписана вредност и кружић (o) за оне колоне у којима може да буде изостављена вредност, тј. у којима може да буде уписана константа *null*. Кључ може да буде: примарни кључ – pk (енгл. Primary key), кључ јединствености – uk (енгл. Unique key) или страни кључ – fk (енгл. Foreign key). Страни кључ ће бити објашњен мало касније приликом објашњења како се мапира веза. Атрибут који је био примарни јединствени идентификатор на ЕР дијаграму, постаје примарни кључ. Остали кандидати за примарни јединстсвени идентификатор постају колоне са ограничењем јединствености. На пример, атрибут *maticni broj* клуба је податак који је јединствен за сваки клуб и два клуба не могу да имају исту вредност матичног броја. У колони која представља тај атрибут у бази, не смеју да постоје две исте вредности јер би то значило да два клуба имају исти матични број.

ПИБ број, назив, број пословног рачуна, број решења и матични број су подаци који су јединствени за сваки клуб. ПИБ број је примерани кључ и означен је са pk. Примарни кључ као ограничење подразумева и кључ јединствености и обавезност постојања податка. Поред назива осталих колона, које имају вредности које морају да буду јединствене за сваку инстанцу, треба да стоји кључ јединствености uk. Иако ретко, постоје особе са истим именом и презименом, тако да може да се деси да се лице на које се шаље пошта за клуб зове исто у два различита клуба. Адреса клуба може да представља адресу пословне зграде у којој више клубова може да има канцеларије, па ни адреса није јединствена. У неким ситуацијама два клуба могу да имају посебан уговор о сарадњи па могу да имају заједнички сајт, а чак и да деле канцеларију па би у том случају број телефона и број факса били исти. Све ове ситуације треба узети у обзир и због њих треба бити обазрив код додељивања кључа јединствености колонама.




| **Назив табеле**: klubovi | | |
|---|---|---|
| **Кључ** | | **Назив колоне** |
| pk | * | pib |
| uk | * | naziv |
| | * | adresa |
| uk | o | br_racuna |
| uk | o | br_resenja |
| uk | o | maticni_br |
| | * | telefon |
| | o | faks |
| | o | mejl |
| | o | sajt |
| | o | adr_delatnosti |
| | * | adr_za_postu |
| | * | lice_za_postu |

Слика 30. Превођење ентитета *Klub* у табелу *Klubovi*

На следећој слици је приказана табела *Klubovi* са унетим подацима о клубовима. Неке колоне су изостављене са слике због прегледности. Колоне *pib*, *naziv* и *telefon* морају да имају вредност за сваки ред, тј. за сваки клуб. У колонама *faks*, *mejl* и *sajt* можемо имати и *null* константу приказану на овој слици цртицом којом можемо обележити недостатак података у ћелији.

| **pib** | **naziv** | **telefon** | **faks** | **mejl** | **sajt** | ... |
|---|---|---|---|---|---|---|
| 101835821 | Cukarica | 0649924768 | - | office@karatecukarica.org.rs | www.karatecukarica.rs | ... |
| 100234561 | Nippon | 0113981759 | 0113981758 | office@bwc-world.org | www.nippon.org.rs | ... |
| 101724710 | Rakovica | 011555777 | - | - | - | |
| 103823126 | Sunce | 0112281634 | - | srksunce@srksunce.rs | www.srksunce.rs | ... |
| 101946932 | Roda | 011888222 | - | - | - | ... |

Слика 31. Подаци у табели *Klubovi*

Интенгритет колоне говори о томе да сви подаци у једној колони морају да имају вредности које су конзистентне са дефинисаним типом података за ту колону и да поштују ограничења дефинисана над том колоном.

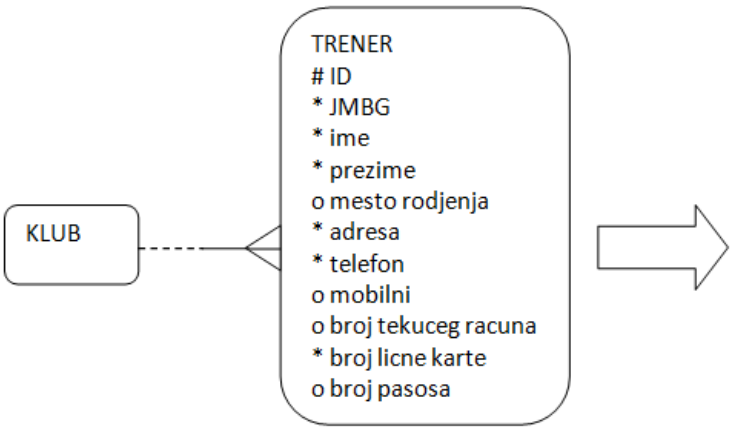
Сви подаци у колони *pib* су бројеви са тачно 9 цифара. Тип података који нам служи за чување бројева се назива NUMBER. Може да се дефинише број цифара. Тип података колоне *pib* ће бити NUMBER(9). Уколико је потребно да чувамо децималне бројеве, може да се прецизира и број децималних места, на пример NUMBER(6,2), где је 6 укупан број цифара, а 2 број

децималних места. За текстулекстуалне податке се користе типови: CHAR, VARCHAR2 и CLOB. Тип CHAR претставља текстуални податак фиксне дужине. Овај тип података бисмо употребили за јединствени матични број грађана који је, иако састављен само од цифара, текстуални податак фиксне дужине од 13 знакова. Као и код бројевног типа, у загради наводимо тачну дужину, на пример CHAR(13). Називи клубова су текстуално подаци различитих дужина. На пример, дужина назива „Чукарица“ је 8 знакова, а дужина назива „Рода“ је 4. Потребно је препознати која је највећа могућа дужина назива клуба и према томе дефинисати тип података за колону *naziv*, на пример VARCHAR2(40). Тип података CLOB (енгл. Character Large Object) служи за јако дугачке текстове. Овим типом би могла да се представи дијагноза код прегледа јер то може да буде и дужи текст са описом, препоруком и закључком лекара. Логичка променљива је променљива која има две могуће вредности тачно (енгл. true) и нетачно (енгл. false). Тип података за логичке променљиве је BOOLEAN. Датуми се у бази података чувају прецизно (дан, месец, година, сат, минут, секунд). Тип података за чување датума и времена је DATE.

Уколико колона не сме да садржи константу *null* за њу се дефинише ограничење NOT NULL. Уколико све вредности у колони морају да буду јединствене, за колону се дефинише ограничење UNIQUE. За колону се дефинише ограничење PRIMARY KEY уколико је она примарни кључ.

Референцијални интегритет се односи на везу међу табелама. Табеле су повезане уколико у једној постоји колона која је страни кључ, и у којој су вредности које упућују на примарни кључ или колону са ограничењем јединствености у другој табели. Све вредности у колони која је страни кључ морају да постоје у колони друге табеле на коју показују или да буду једнаке константи *null* уколико је то дозвољено за ту колону.

Свака веза са дијаграма ће постати колона која се назива страни кључ. Уколико је веза типа 1:M, колона се додаје у ону табелу која одговара ентитету на страни везе на којој се налази ознака за више. Можемо да посматрамо везу између ентитета *Trener* и *Klub*. У сваком клубу може да ради један или више тренера. Ознака за више је на страни везе ближе ентитету *Trener*, па ће у табели *Treneri* постојати додатна колона *pib_kluba* која ће садржати податак о томе у ком клубу ради тренер. Тренер мора да ради у неком клубу, па је ова веза обавезна за сваког тренера и колона *pib_kluba* мора да има вредност у сваком реду. То се обележава звездицом. У случајевима када је крај везе на страни више испрекидана линија, за колону која је страни кључ ставио би се кружић, тј. ознака за опционалност.



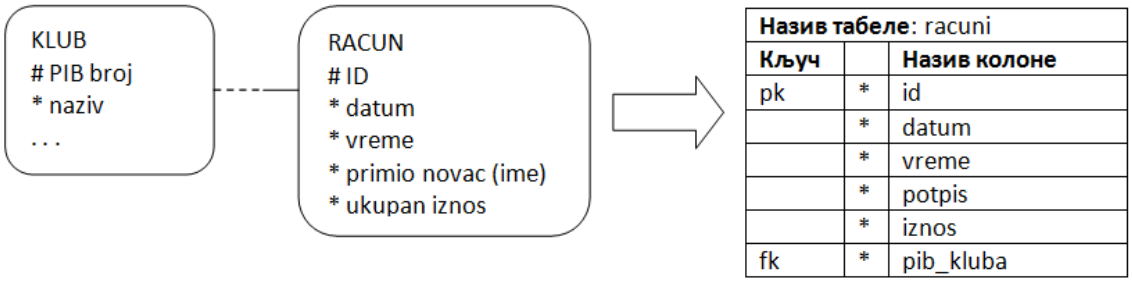

| **Назив табеле**: treneri | | |
|---|---|---|
| **Кључ** | | **Назив колоне** |
| pk | * | id |
| uk | * | jmbg |
| | * | ime |
| | * | prezime |
| | o | mesto_rodj |
| | * | adresa |
| | * | telefon |
| uk | o | mobilni |
| uk | o | br_racuna |
| uk | * | licna_karta |
| uk | o | pasos |
| fk | * | pib_kluba |

**Слика 32. Превођење ентитета *Trener* и везе између ентитета *Klub* и *Trener* у табелу *Treneri***

Како у једном клубу ради више тренера, било би немогуће уз податке о једном клубу чувати шифре (идентификационе бројеве) свих тренера који раде у њему. Међутим, сваки тренер ради тачно у једном клубу и поред свих података о њему чувамо и ПИБ клуба за који ради. Када се прочита ПИБ клуба, који је уписан уз податке о тренеру у табели *Treneri*, у табели *Klubovi* може да се пронађе одговарајући број у колони која је примарни кључ, па самим тим и сви остали подаци о клубу у којем тај тренер ради. Ово је илустровано следећом сликом. Подаци су преузети са [7].


Слика 33. Табеле *Treneri* и *Klubovi* повезане страним кључем

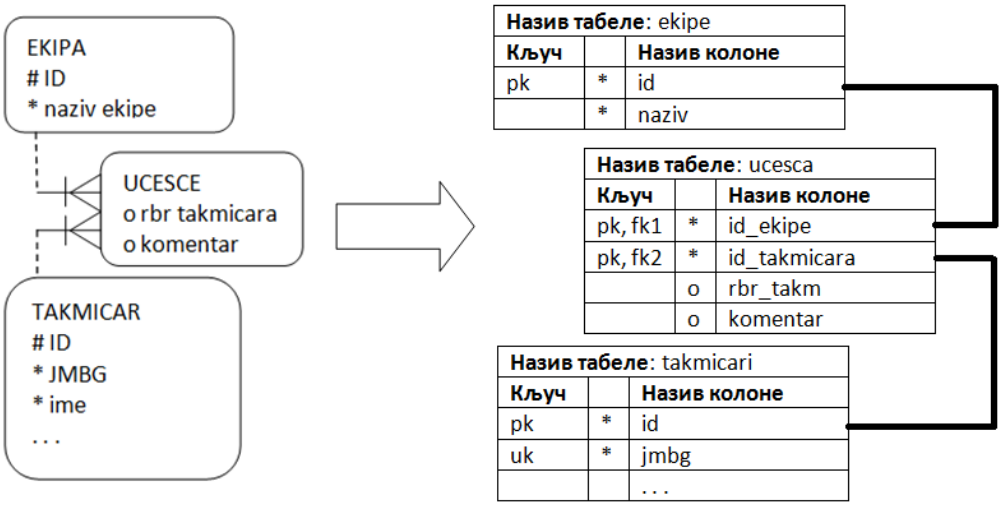
Уколико је веза типа 1:1 обавезна са једне стране (пуна линија), страни кључ се као колона додаје у табелу са те стране. На пример, клубу може да се изда један и само један рачун, али рачун мора да се изда једном и тачно једном клубу. Страни кључ ће се додати у табелу *Racuni* како је и приказано на следећој слици.




Слика 34. Превођење везе типа 1:1

У општем случају када је веза типа 1:1 опциона целом дужином, свеједно је у коју таблу додајемо страни кључ. Међутим, важно је анализирати конкретну ситуацију и на основу тога донети одлуку где је згодније и природније додати страни кључ.

Веза типа M:M између два ентитета се разрешава додавањем новог ентитета, односно табеле у бази података. Примарни кључ те табеле, је сложен и састоји се од примарних кључева почетних табела. Делови тог примарног кључа су уједно и страни кључеви коју показују на две почетне табеле. Чим у табели има више од једног страног кључа, потребно је увести нумерацију страних кључева, па се они обележавају са fk1, fk2, fk3 и тако редом. Посматрајмо таблу *Ucesca* која се налази уместо везе типа M:M између табела *Ekipe* и *Takmicari*. Примарни кључ табеле *Ucesca* је сложен и састоји се од комбинације вредности две колоне, *id_ekipe* и *id_takmicara*.




Слика 35. Превођење везе типа M:M

Постоји и друга могућност, а то је да се уведе вештачки примарни кључ за табелу која је настала као разрешење везе типа M:M, у овом примеру за табелу *Ucesca*. У том случају, колоне *id_ekipe* и *id_takmicara* су и даље страни кључеви ка табелама *Ekipe* и *Takmicari*, али нису више део сложеног примарног кључа.

| Назив табеле: ucesca | | |
|---|---|---|
| **Кључ** | | **Назив колоне** |
| pk | * | id |
| fk1 | * | id_ekipe |
| fk2 | * | id_takmicara |
| | o | rbr_takm |
| | o | komentar |

Слика 36. Табела *Ucesca*

Пример ових табела са једним малим делом уписаних података је приказан на следећој слици. Неке колоне су изостављене због прегледности.

**ekipe**

| id | naziv |
|---|---|
| 1 | Prvi tim - Cukarica |
| 2 | Drugi tim - Cukarica |

**takmicari**

| id | ime | prezime | . . . |
|---|---|---|---|
| 1 | Milica | Zoranovic | . . . |
| 2 | Olivera | Stosic | . . . |
| 3 | Lena | Kulic | |
| 4 | Branislav | Zoranovic | . . . |
| 5 | Ana | Milisavljevic | . . . |

**ucesca**

| id | id_ekipe | id_takmicara | rbr_takm | komentar |
|---|---|---|---|---|
| 1 | 1 | 4 | 5 | kapiten |
| 2 | 2 | 5 | 6 | rezerva |
| 3 | 2 | 3 | 3 | - |
| 4 | 2 | 1 | 1 | kapiten |
| 5 | 2 | 2 | 4 | - |

Слика 37. Подаци у повезаним табелама *Ekipe*, *Takmicari* и *Ucesca*

Постоје и ограничења која је дефинисао будући корисник, односно која су произашла из правила пословања за које се креира база.

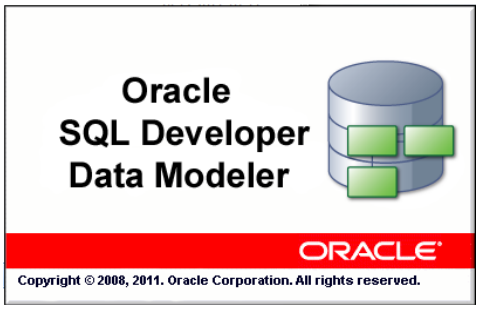
Свака категорија има податке или о појасу или о тежини такмичара. На дијаграму су сви ови атрибути опциони, јер уколико је категорија у катама не постоји податак о тежини, док за категорије у борбама не постоји податак о појасу. Ово значи да у бази података једна од колона *min_pojas, max_pojas, min_kg* и *max_kg* мора да има вредност. Уколико бар једна од колона *min_pojas* и *max_pojas* има вредност, онда *min_kg* и *max_kg* немају, и обратно. Уводи се ограничење које ће то да прецизира:

```
CHECK CONSTRAINT
((min_pojas IS NOT NULL OR max_pojas IS NOT NULL) AND  min_kg IS NULL AND max_kg IS NULL) OR
((min_kg IS NOT NULL OR max_kg IS NOT NULL) AND min_pojas IS NULL AND max_pojas IS NULL)
```

Постоје и друга кориснички дефинисана ограничења која се односе на саме уписане податке. На пример, ЈМБГ је текстуални податак од 13 цифара. Првих седам цифара представају датум и то: две цифре за редни број дана, две цифре за редни број месеца и три цифре за годину. У том смислу ЈМБГ не може да почне вредностима које не могу да означавају датум. На пример: 3524001777111 сигурно није валидан ЈМБГ јер не постоји дан 35 у месецу 24.

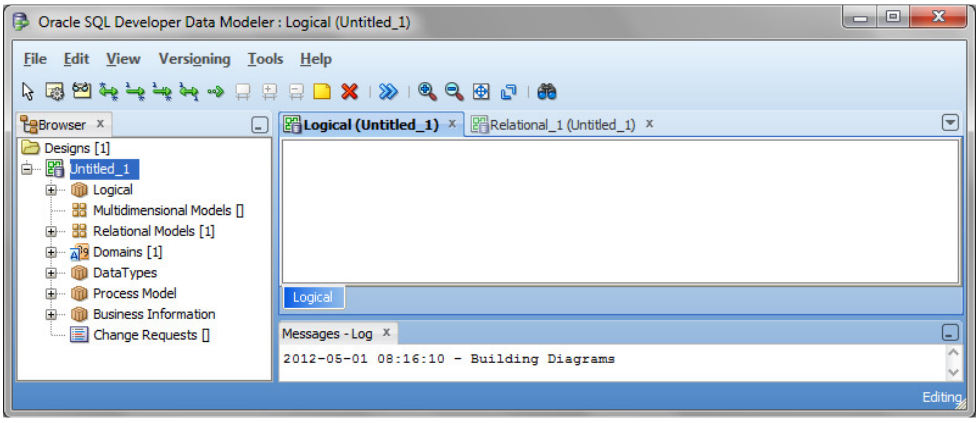
# 6. Софтверски алат Oracle Data Modeler

Један од софтверских алата којим можемо да цртамо дијаграме и правимо припрему модела за изградњу базе података је **Oracle Data Modeler**. Овај алат може да се преузме са сајта www.oracle.com компаније Oracle [2].




Слика 38. Oracle Data Modeler

Након покретања софтвера, појављује се простор за креирање новог логичког модела *Logical (Untitled_1)*. Испод падајућег менија се налази линија са расположивим алаткама које укључују: дугме за селекцију, дугмад за креирање новог ентитета, веза, лукова, дугме за уклањање елемента са дијаграма, дугме за превођење у релациони модел и помоћну дугмад.

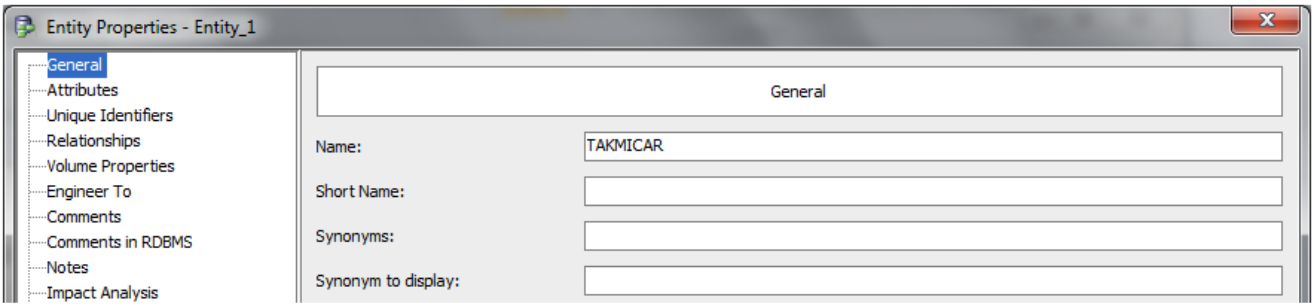



Слика 39. Алат за креирање логичког модела

Након притиска на дугме *New Entity*, можемо да поставимо ентитет на површину за цртање дијаграма тако што тастер миша притиснемо у горњем левом углу а отпустимо у доњем десном углу правоугаоне површине у којој ће након тога бити исцртан ентитет.

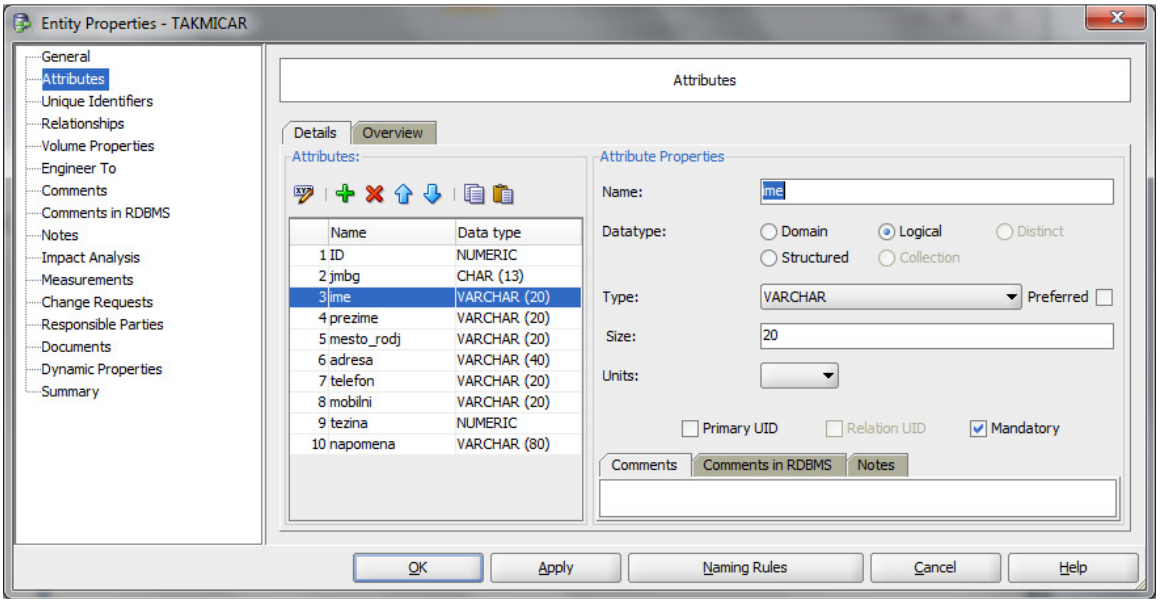

Слика 40. Нови ентитет

Појављује се прозор са свим опцијама за дефинисање ентитета. Први ентитет који додајемо на дијаграм је ентитет *Takmicar*. Уместо вредности *Entity_1*, потребно је унети назив ентитета у поље *Name* на језичку *General* отвореног прозора.



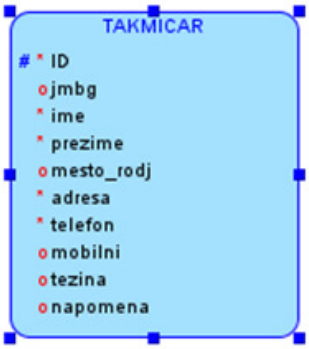

Слика 41. Додавање новог ентитета моделу

Следи формирање листе атрибута. Нови атрибут се додаје на листу притиском на зелени знак плус. За атрибут се попуни назив (енгл. Name) и са листе се изабере тип података (енгл. Type). Уколико је атрибут обавезан потребно је да се штиклира *Mandatory*, а уколико је примарни јединствени идентификатор *Primary UID*.




Слика 42. Додавање атрибута ентитету

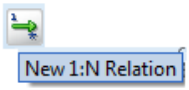
Након притиска на дугме OK, појављује се ентитет на дијаграму. Претходни прозор може поново да се отвори избором ставке *Properties* након притиска на десни тастер миша над ентитетом.




Слика 43. Ентитет *Takmicar*

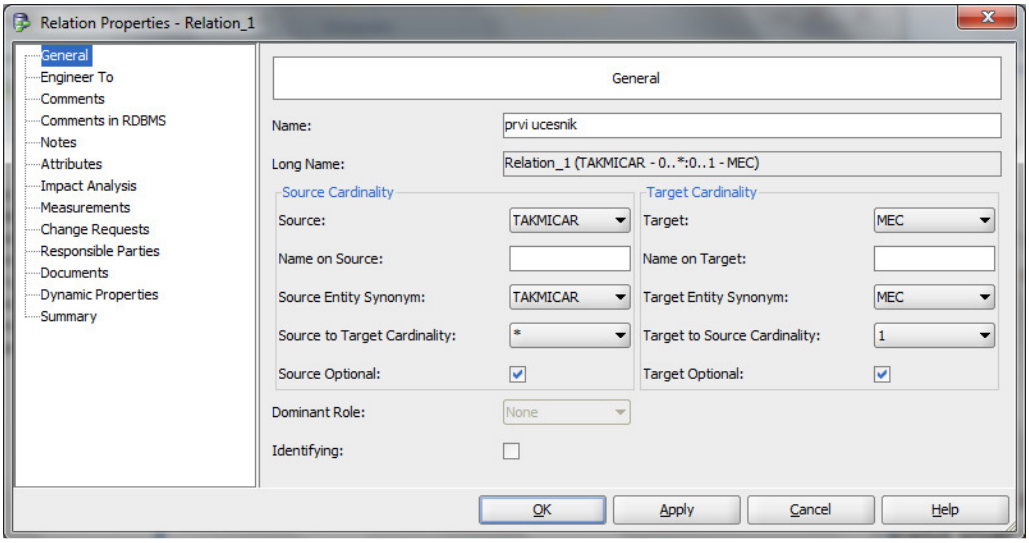
Истим поступком се креира и ентитет *Mec*. Након креирања ентитета потребно је да се креирају и везе које их спајају.

Након притиска на дугме *New 1:N Relation* може да се повуче веза од једног до другог ентитета на дијаграму тако што се тастером миша прво притисне један ентитет, а затим се притисне други. У овом примеру прво се кликне на ентитет *Takmicar*, а затим *Mec*.



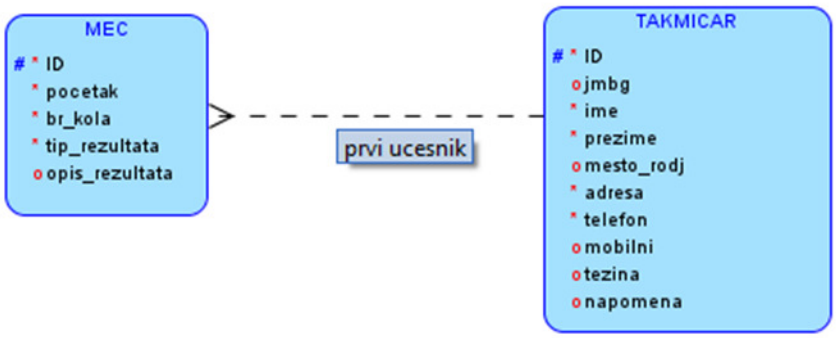

Слика 44. Нова веза типа 1:M

Појављује се прозор у којем се дефинише веза тако што јој се додели име и потврди се који је први, а који је други ентитет који учествује у вези. Први би требао да буде онај код којег је ознака за кардиналност један, а други онај код којег је ознака за кардиналност више. У овом примеру, сваки такмичар може да буде први учесник у једном или више мечева, па је први ентитет *Takmicar*, а други *Mec*.



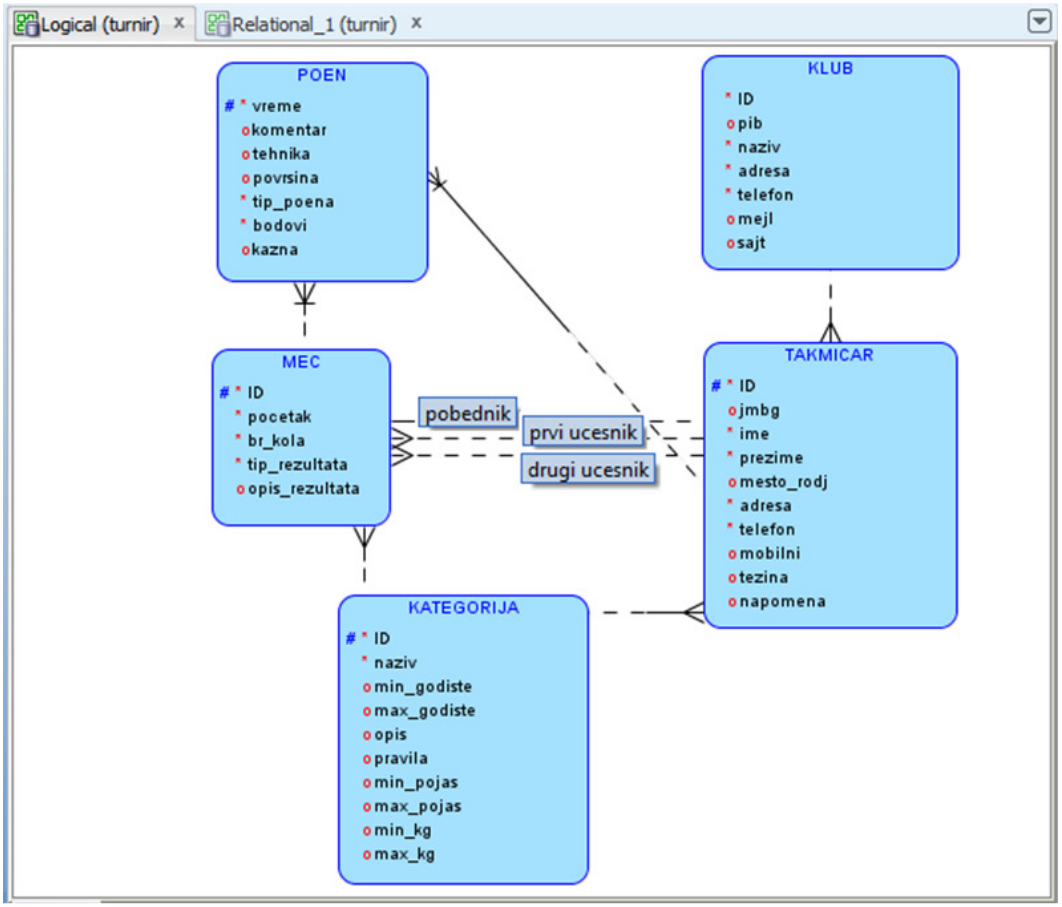

Слика 45. Креирање везе између ентитета *Takmicar* и *Mec*

Уколико треба да променимо опционалност везе, треба да кликнемо поред *Source Optional* или *Target Optional*. Оба су на почетку штиклирана, што значи да је на почетку веза целом дужином опциона. Након притиска на дугме *OK* веза се појављује на дијаграму.



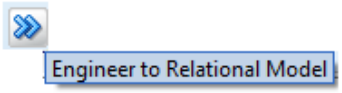

Слика 46. Веза између ентитета *Takmicar* и *Mec*

Дијаграм ентитета и веза турнира са пет ентитета је приказан на следећој слици.



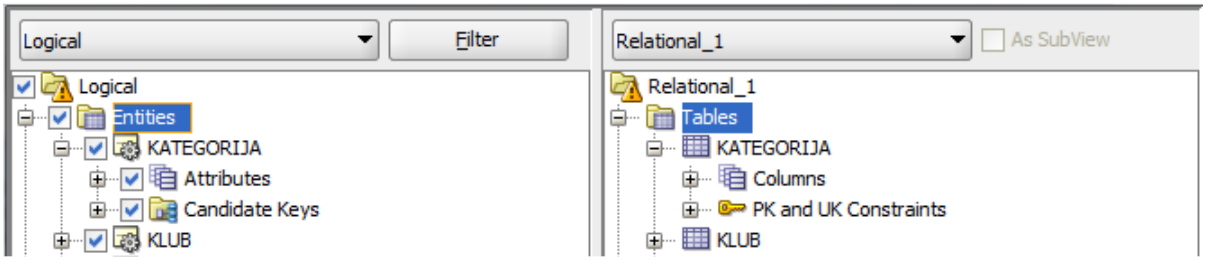

Слика 47. Дијаграм ентитета и веза

Кликом на дугме *Engineer to Relational Model* се покреће поступак аутоматске припреме логичког модела за израду базе података.




Слика 48. Превођење у релациони модел

Појављује се прозор у којем може да се изврши упоређивање појмова са логичког модела и њима одговарајућих појмова физичког модела. На пример, сваки ентитет ће постати табела.




Слика 49. Превођење у релациони модел

Након притиска на дугме *Engineer*, добија се релациони модел турнирског такмичења са пет табела. Ознаке на аутоматски генерисаном моделу се мало разликују од приказане теорије мапирања, па ће та слика овде бити изостављена и биће приказано мапирање у складу са објашњењима и правилима изложеним у овом раду.

**Назив табеле**: poeni

| Кључ | | Назив колоне | Тип података |
|---|---|---|---|
| pk | * | id | number |
| | * | vreme | number |
| | o | komenar | varchar2 |
| | o | tehnika | varchar2 |
| | o | povrsina | varchar2 |
| | * | tip_poena | varchar2 |
| | * | bodovi | number |
| | o | kazna | boolean |
| fk1 | * | id_takm | number |
| fk2 | * | id_meca | number |

**Назив табеле**: klubovi

| Кључ | | Назив колоне | Тип података |
|---|---|---|---|
| pk | * | id | number |
| uk | * | naziv | varchar2 |
| | * | adresa | varchar2 |
| | * | telefon | varchar2 |
| | o | mejl | varchar2 |
| | o | sajt | varchar2 |

**Назив табеле**: takmicari

| Кључ | | Назив колоне | Тип података |
|---|---|---|---|
| pk | * | id | number |
| uk | o | jmbg | char |
| | * | ime | varchar2 |
| | * | prezime | varchar2 |
| | o | mesto_rodj | varchar2 |
| | * | adresa | varchar2 |
| | * | telefon | varchar2 |
| | o | mobilni | varchar2 |
| | o | tezina | number |
| | o | napomena | varchar2 |
| fk1 | * | id_kluba | number |
| fk2 | * | id_kategorije | number |

**Назив табеле**: mecevi

| Кључ | | Назив колоне | Тип података |
|---|---|---|---|
| pk | * | id | number |
| | * | pocetak | date |
| | * | br_kola | number |
| | * | tip_rezultata | varchar2 |
| | o | opis_rezultata | varchar2 |
| fk1 | o | id_prvog | number |
| fk2 | o | id_drugog | number |
| fk3 | o | id_pobednika | number |
| fk4 | * | id_kateogrije | number |

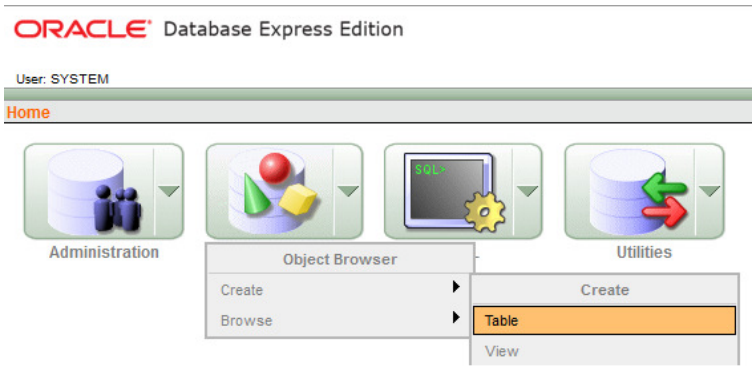
**Назив табеле**: kategorije

| Кључ | | Назив колоне | Тип података |
|---|---|---|---|
| pk | * | id | number |
| uk | * | naziv | varchar2 |
| | o | min_godiste | number |
| | o | max_godiste | number |
| | o | opis | varchar2 |
| | o | pravila | varchar2 |
| | o | min_pojas | varchar2 |
| | o | max_pojas | varchar2 |
| | o | min_kg | number |
| | o | max_kg | number |

Слика 50. Превођење из логичког у физички модел

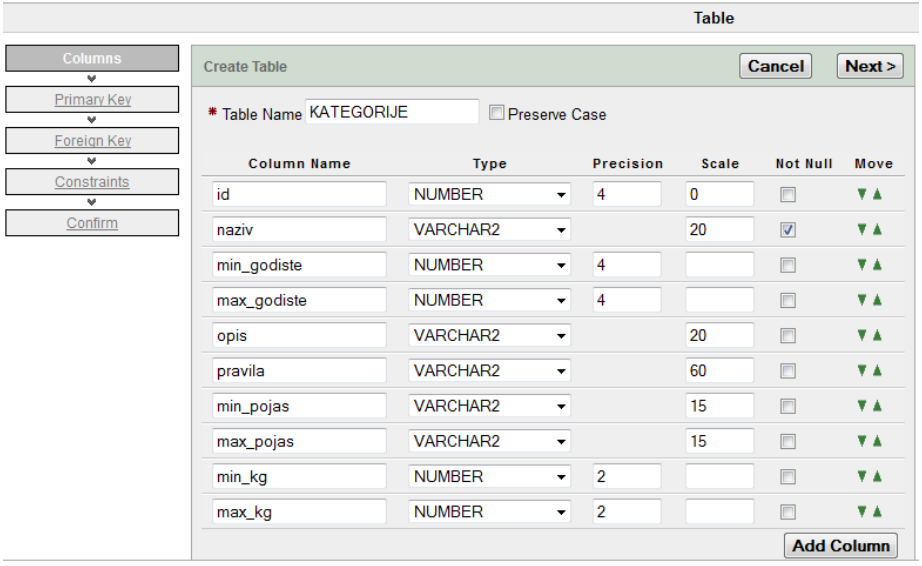
## 7. Креирање базе података турнирског такмичења

Са сајта компаније Oracle [2] може да се преузме програм **Oracle Database 10g Express Edition**. То је систем за управљање базама података (енгл. Database Management System). Релациона база података се састоји од више међусобно повезаних табела. Уколико једна табела има страни кључ који упућује на другу табелу, потребно је водити рачуна о редоследу креирања. На пример, табела *Takmicari* има страни кључ *id_kategorije* који упућује на табелу *Kategorije*. Прво мора да се креира табела *Kategorije*, а затим табела *Takmicari*. Прво могу да се креирају све табеле које немају стране кључеве. Табела се креира избором *Create -> Table* из одељка *Object Browser*.



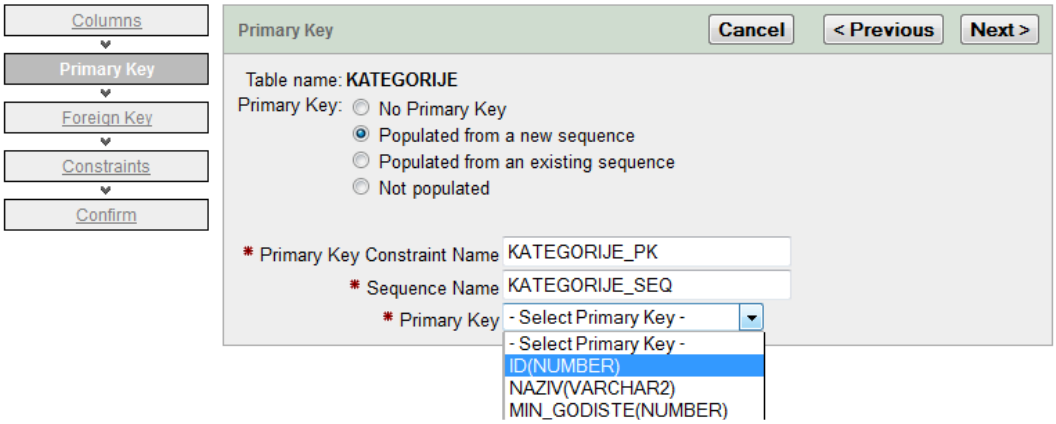

Слика 51. Oracle Database 10g Express Edition

Попуни се име табеле (енгл. Table Name). Када се прозор отвори појављује се простор за унос података о осам колона. Уколико табела има више колона, притиском на дугме *Add Column* се појављује простор за додавање података о још једној колони. Свакој колони се додели име и тип података. За бројевни тип података NUMBER се прецизира укупан број цифара (енгл. Precision) и колико од тих цифара су децимална места (енгл. Scale). Уколико се изостави да се упише број децималних места или се упише нула, бројеви који се чувају у тој колони ће бити цели бројеви. За текстуални тип VARCHAR2 се прецизира највећа дужина изражена бројем знакова. За колоне које морају да имају податак у сваком реду штиклира се поље *Not Null*.



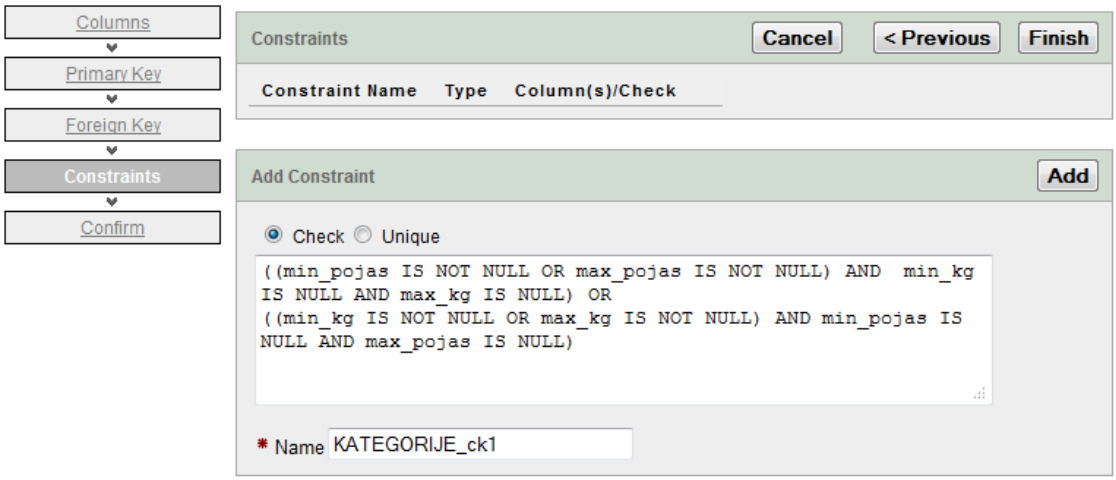

Слика 52. Дефинисање колона табеле *Kategorije*

Притиском на дугме *Next* појављује се прозор за постављање примарног кључа. Потребно је да се притисне избор *Populate from a new sequence*. Након тога може да се изабере колона која ће бити примарни кључ са списка свих колона додељених табели. Секвенца је објекат базе података који служи за генерисање вредности примарног кључа. За вредности примарног кључа је важно да све буду различите. Употребом секвенце, обезбеђује се да се приликом уношења података у таблу уносе само вредности које до тог тренутка нису унете у колону која је примарни кључ. Систем аутоматски додели назив ограничењу као комбинацију назива табеле и скраћенице која се односи на само ограничење. Скраћенице су: pk за примарни кључ, fk за страни кључ, uk за кључ јединствености и nn за ограничење Not null. Колона *id* је примарни кључ табеле *Kategorije* па је назив ограничења KATEGORIJE_PK. Назив секвенце која ће да генерише вредности за примарни кључ је KATEGORIJE_SEQ.



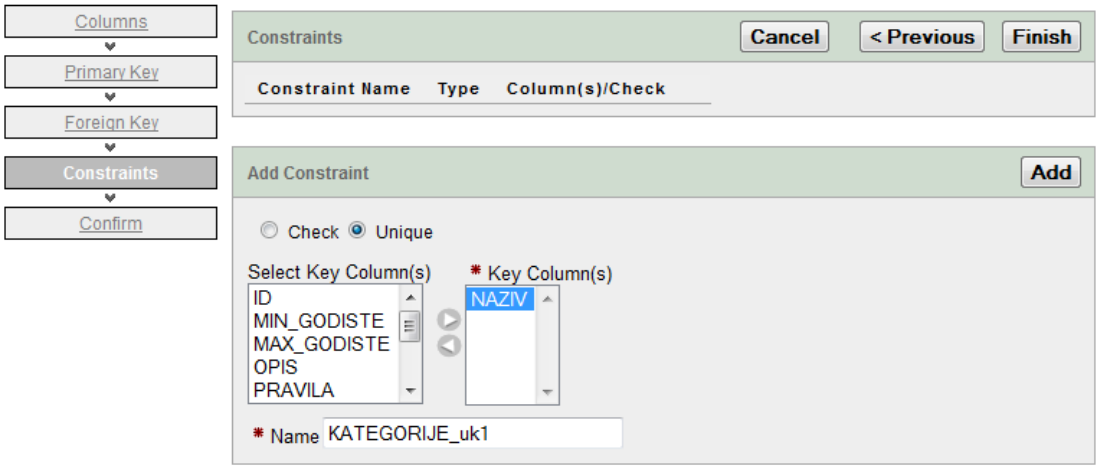

Слика 53. Дефинисање примарног кључа

Притиском на дугме *Next* два пута прескаче се прозор за дефинисање страних кључева јер их ова табела нема, и стиже се до прозора за дефинисање других ограничења. Ова табела има једно ограничење типа Check. То ограничење је детаљно објашњено раније у тексту. Овде га наводимо. Притиском на дугме *Add* ограничење се додаје табели.



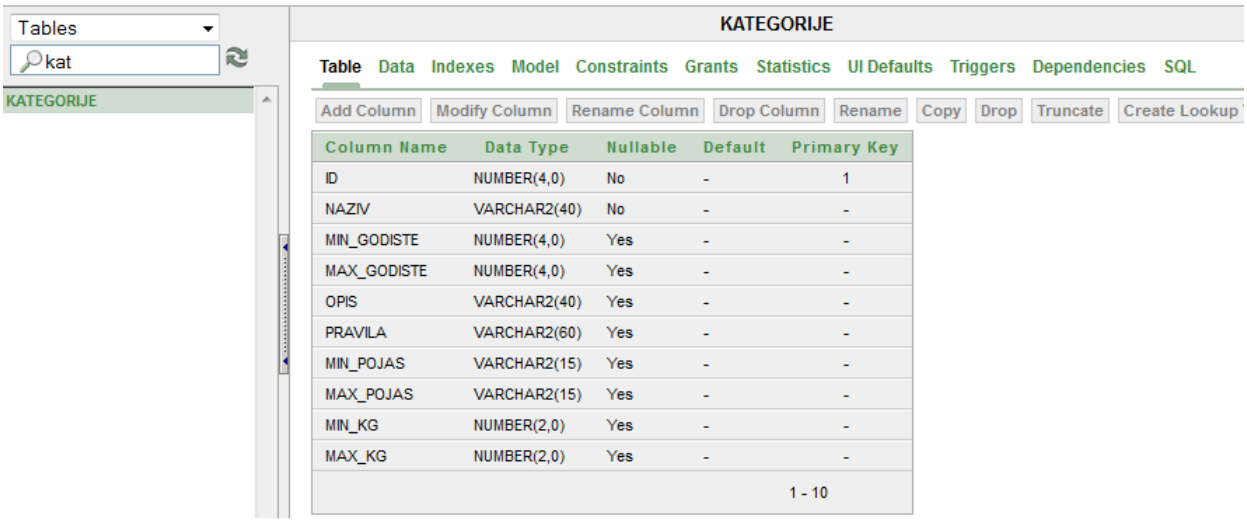

Слика 54. Дефинисање ограничења

Селекцијом избора *Unique* могу да се додају сва ограничења јединствености. У овој табели колона *naziv* мора да има јединствене вредности, па се након њене селекције у листи *Select Key Column(s)* притисне стрелица на десно да би се колона пребацила у листу *Key Column(s).*




Слика 55. Дефинисање ограничења

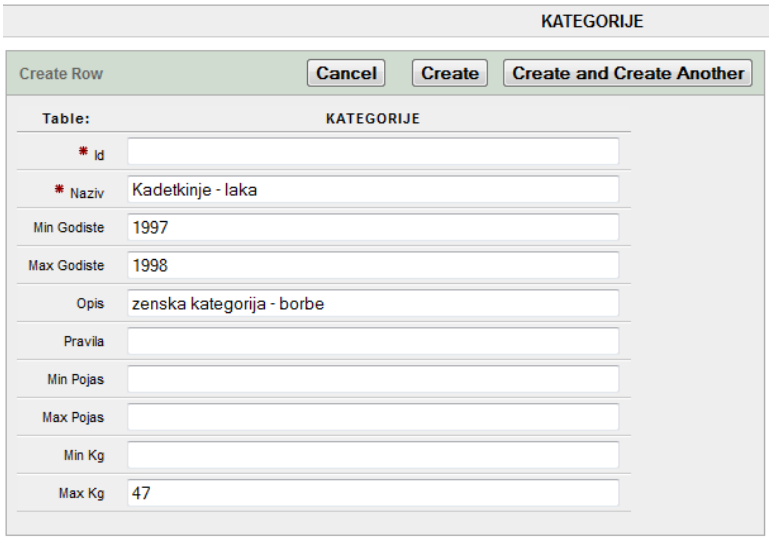
Ограничења добијају аутоматски називе у складу за конвенцијом именовања која је раније објашњења. Уколико је потребно да се додају још нека ограничења ових типова, притисне се дугме *Add*. Након притиска на дугме *Finish* и потврде на дугме *Create*, табела ће бити креирана.




Слика 56. Табела *Kategorije*

У одељку *Object Browser* може да се изабере табела из списка свих табела. Под језичком *Table* постоје могућности да се структура табеле измени. Притиском на одговарајуће дугме може да се дода колона (енгл. Add Column), да се измени дефиниција колоне (енгл. Modify Column), да се преименује колона (енгл. Rename Column), да се уклони колона (енгл. Drop Column), да се преименује цела табела (енгл. Rename) и да се уклони цела табела (енгл. Drop).

Да би се приступило подацима у табели потребно је да се пређе на језичак *Data*. Табела која је тек креирана не садржи податке. Подаци могу да почну да се уносе када се притисне на дугме *Insert Row*. Уноси се један по један ред. Подаци које немамо могу да се изоставе само у оним колонама где је дозвољена вредност null. Може да се изостави и вредност примарног кључа јер ће она бити аутоматски попуњена из одговарајуће секвенце.




Слика 57. Додавање података у табелу *Kategorije*

Табела попуњена подацима је приказана на следећој слици. Подаци су преузети са [8].

Data Indexes Model Constraints Grants Statistics UI Defaults Triggers Dependencies SQL

Count Rows Insert Row

| ID | NAZIV | MIN_GODISTE | MAX_GODISTE | OPIS | PRAVILA | MIN_POJAS | MAX_POJAS | MIN_KG | MAX_KG |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Kadetkinje - laka | 1997 | 1998 | zenska kategorija - borbe | - | - | - | - | 47 |
| 2 | Kadetkinje - srednja | 1997 | 1998 | zenska kategorija - borbe | - | - | - | - | 54 |
| 3 | Kadetkinje - teska | 1997 | 1998 | zenska kategorija - borbe | - | - | - | 54 | - |
| 4 | Kadeti - super laka | 1997 | 1998 | muska kategorija - borbe | - | - | - | - | 52 |
| 5 | Kadeti - laka | 1997 | 1998 | muska kategorija - borbe | - | - | - | - | 57 |
| 6 | Kadeti - srednja | 1997 | 1998 | muska kategorija - borbe | - | - | - | - | 63 |
| 7 | Kadeti - teska | 1997 | 1998 | muska kategorija - borbe | - | - | - | - | 70 |
| 8 | Kadeti - super teska | 1997 | 1998 | muska kategorija - borbe | - | - | - | 70 | - |
| 9 | Kadetkinje - nizi pojasevi | 1997 | 1998 | zenska kategorija - kate | - | - | oranz | - | - |
| 10 | Kadetkinje - srednji pojasevi | 1997 | 1998 | zenska kategorija - kate | - | zeleni | plavi | - | - |
| 11 | Kadetkinje | 1997 | 1998 | zenska kategorija - kate | - | - | crni | - | - |

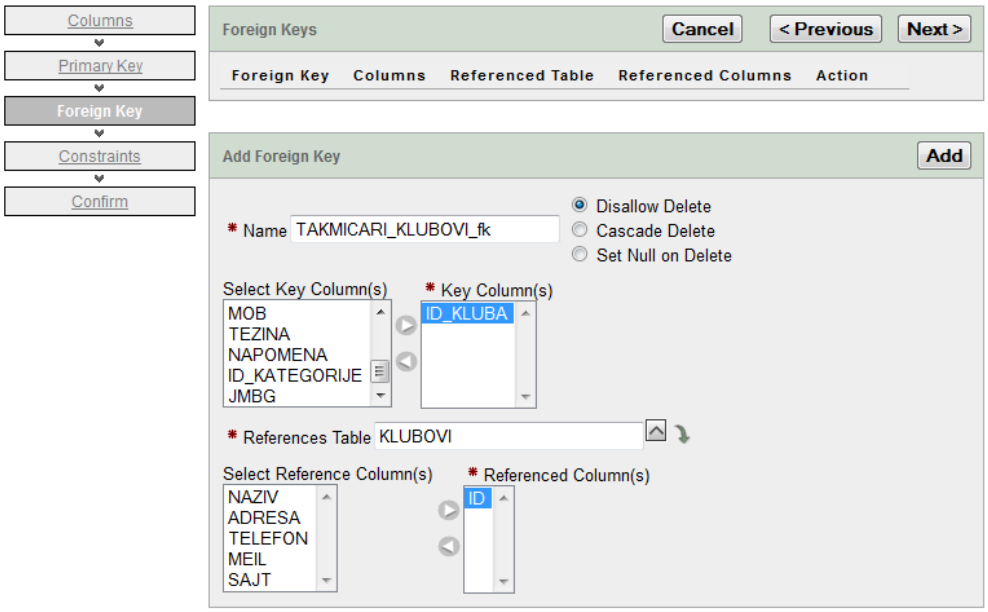
Слика 58. Подаци у табели *Kategorije*

На исти начин се креира и попуни подацима табела *Klubovi*, са колоном *id* која је примарни кључ, обавезним колонама *naziv*, *adresa* и *telefon*, и опционим колонама *meil* и *sajt*.

Као следећа може да се креира табела *Takmicari*. Ова табела има два страна кључа који је повезују са табелама *Klubovi* и *Kategorije*. Сви кораци у креирању ове табеле су исти као и за претходне две табеле, али је потребно обратити пажњу на још један корак који нисмо имали раније, а то је дефинисање страних кључева.

За сваки нови страни кључ, окружење ће да понуди назив. Како ова табела има два страна кључа ка две различите табеле, добро је преименовати ограничење тако да се из његовог назива можемо јасно прочитати у којој табели се налази страни кључ (назив ове табеле је на првом месту у називу ограничења, и у овом примеру је то табела *Takmicari*) и на коју табелу

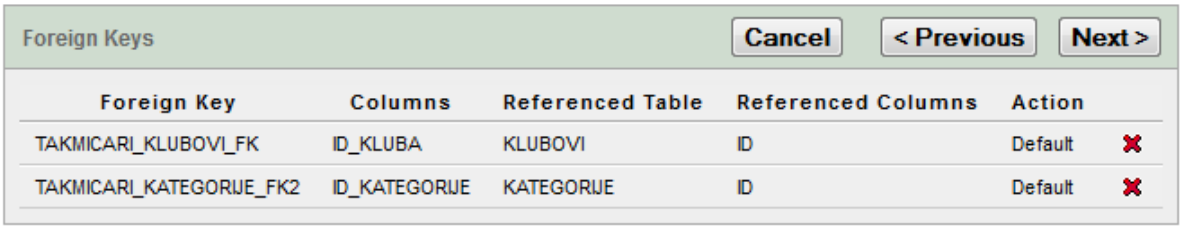
показује (назив ове табеле је други по реду, и у овом примеру је то табела *Klubovi*). На крају назива ограничења стоји скраћеница типа ограничења у овом случају *fk* (енгл. Foreign Key). У пољу *Refereces Table* се изабере табела на коју показује страни кључ и изабере се одговарајућа колона те табеле. На следећој слици је приказана колона *id_kluba* табеле *Takmicari* која треба да буде страни кључ који показује на колону *id* табеле *Klubovi*.




Слика 59. Дефинисање страног кључа

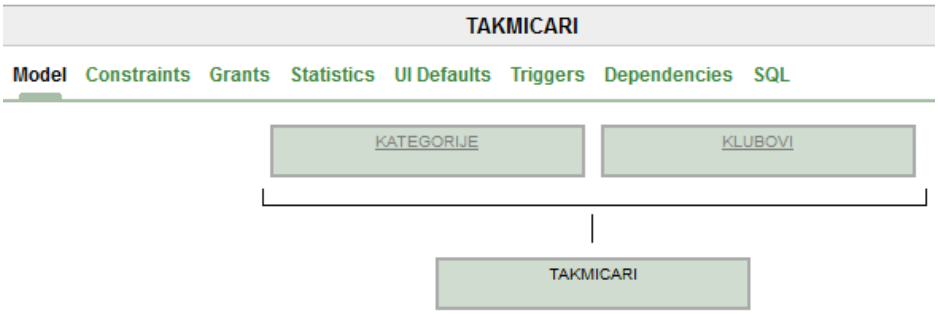
Као вредност страног кључа у табели *Takmicari* стоје вредности примарног кључа из табеле *Klubovi*. Приликом креирања страног кључа потребно је и одредити каква акција треба да се предузме уколико се покуша брисање из табеле *Klubovi* клуба који има такмичаре у табели *Takmicari*. Уколико се изабере опција *Disallow Delete* брисање неће бити могуће. Друге две могућности су да се, уколико се обрише клуб из табеле *Klubovi* обришу и сви његови такмичари (каскадно брисање, тј. опција *Cascade Delete*) или да се вредност страног кључа у табели *Takmicari* постави на *null* у оним редовима који су показивали на обрисан клуб (опција *Set Null on Delete*).

На описани начин се креира и други страни кључ и оба могу да се виде на списку страних кључева.




Слика 60. Страни кључеви у табели *Takmicari*

Након креирања табеле *Takmicari* под језичком *Model* са графикона може да се види са којим је табелама повезана.




Слика 61. Повезане табеле *Kategorije*, *Klubovi* и *Takmicari*

Под језичком Constraints се налази списак свих ограничења дефинисаних над табелом. Нека ограничења имају смислена имена, остала (углавном Not Null) имају системски додељене називе.

TAKMICARI

Table | Data | Indexes | Model | **Constraints** | Grants | Statistics | UI Defaults | Triggers | Dependencies | SQL

Create | Drop | Enable | Disable

| Constraint | Type | Table | Search Condition | Delete Rule | Status | Last Change | Index | Invalid |
|---|---|---|---|---|---|---|---|---|
| SYS_C004099 | C | TAKMICARI | "IME" IS NOT NULL | - | ENABLED | 16.05.2012 | - | - |
| SYS_C004100 | C | TAKMICARI | "PREZIME" IS NOT NULL | - | ENABLED | 16.05.2012 | - | - |
| SYS_C004101 | C | TAKMICARI | "ADRESA" IS NOT NULL | - | ENABLED | 16.05.2012 | - | - |
| SYS_C004102 | C | TAKMICARI | "TELEFON" IS NOT NULL | - | ENABLED | 16.05.2012 | - | - |
| SYS_C004103 | C | TAKMICARI | "ID_KLUBA" IS NOT NULL | - | ENABLED | 16.05.2012 | - | - |
| SYS_C004104 | C | TAKMICARI | "ID_KATEGORIJE" IS NOT NULL | - | ENABLED | 16.05.2012 | - | - |
| TAKMICARI_PK | P | TAKMICARI | - | - | ENABLED | 16.05.2012 | TAKMICARI_PK | - |
| TAKMICARI_KLUBOVI_FK | R | TAKMICARI | - | NO ACTION | ENABLED | 16.05.2012 | - | - |
| TAKMICARI_KATEGORIJE_FK2 | R | TAKMICARI | - | NO ACTION | ENABLED | 16.05.2012 | - | - |
| TAKMICARI_UK1 | U | TAKMICARI | - | - | ENABLED | 16.05.2012 | TAKMICARI_UK1 | - |

1 - 10

Слика 62. Сва ограничења табеле *Takmicari*

Вредности у колони која је страни кључ морају да постоје у табели на коју кључ показује. Уколико се покуша са уносом неке вредности која не постоји, долази до грешке и нови ред неће моћи да се унесе у табелу. На следећој слици је порука која је приказана након покушаја да се унесу подаци о такмичару и да се за шифру категорије наведе број који не постоји као шифра у табели *Kategorije*.

error ORA-02291: integrity constraint (SYSTEM.TAKMICARI_KATEGORIJE_FK2) violated - parent key not found

Слика 63. Вредности у колони која је страни кључ морају да постоје у табели на коју кључ показује

Са објектима у бази података може да се управља и помоћу исказа упитног језика SQL([1]). Постоји група исказа којима се креирају, модификују и бришу објекти (табеле, секвенце...) у бази. Ти искази чине групу исказа Језика за дефиницију података – DDL (енгл. Data Definition Language). Искази којима извлачимо информације из базе, уносимо, бришемо и мењамо податке у бази припадају Језику за манипулацију подацима – DML (енгл. Data Manipulation Language). Поред ових исказа постоје и искази за управљање трансакцијама – TCL (енгл. Transaction Control

Language), као и искази који служе за контролу приступа објектима базе – DCL (енгл. Data Control Language).

| | |
|---|---|
| DML (Data Manipulation Language) | SELECT, INSERT, UPDATE, DELETE, MERGE |
| DDL (Data Definition Language) | CREATE, ALTER, DROP, RENAME, TRUNCATE |
| TCL (Transaction Control Language) | COMMIT, ROLLBACK, SAVEPOINT |
| DCL (Data Control Language) | GRANT, REVOKE |

У овом раду ће бити приказани неки од исказа за управљање објектима и подацима у бази података, тј. неки од исказа из група DDL и DML исказа.

Искази упитног језика SQL се уносе у радни простор који се покреће избором *SQL Commands -> Enter Command* из одељка *SQL*. Исказ, тј. наредба, се покреће након уноса притиском на дугме *Run*, након чега се приказује резултат извршене наредбе или порука о успешности извршења.


Слика 64. Радни простор за унос исказа упитног језика SQL

Табеле у бази података се креирају помоћу наредбе CREATE TABLE. Упитни језик SQL није осетљив на велика и мала слова. У наредби мора да се наведе назив табеле и списак свих колана раздвојених зарезом. За сваку колону се наводи назив, тип података и евентуална ограничења. Сва ограничења у следећем примеру су именована и дефинисана на нивоу колоне. Именовање ограничења је опционо. Уколико се ограничење не именује биће му додељен системски назив. Тип података DATE чува дан, месец, годину, сат, минут и секунд.

Следећа наредба креира табелу *Mecevi*:

```
CREATE TABLE MECEVI
(id NUMBER(6) CONSTRAINT mecevi_pk PRIMARY KEY,
pocetak DATE NOT NULL,
br_kola NUMBER(2) NOT NULL,
tip_rezultata VARCHAR2(20) NOT NULL,
opis_rezultata VARCHAR2(150),
id_prvog NUMBER(4) CONSTRAINT mecevi_prvi_fk REFERENCES TAKMICARI(id),
id_drugog NUMBER(4) CONSTRAINT mecevi_drugi_fk REFERENCES TAKMICARI(id),
id_pobednika NUMBER(4) CONSTRAINT mecevi_pobednik_fk REFERENCES TAKMICARI(id),
id_kategorije NUMBER(4) CONSTRAINT mecevi_kat_fk REFERENCES KATEGORIJE(id) NOT NULL
)
```

Ограничења (осим ограничења NOT NULL) могу да се дефинишу и на нивоу целе табеле тако што се поброје после дефиниције последње колоне. Следећом наредбом би такође могла да се

креира табела *Mecevi*. Сва ограничења су иста. Разлика је у томе што су у овој наредби раздовјене дефиниције колона и дефиниције ограничења која су побројана на самом крају. Како су сада ограничења дефинисана на нивоу целе табеле, потребно је да се наведу у заградама и називи колона на које се односе.

```
CREATE TABLE MECEVI
(id NUMBER(6),
pocetak DATE NOT NULL,
br_kola NUMBER(2) NOT NULL,
tip_rezultata VARCHAR2(20) NOT NULL,
opis_rezultata VARCHAR2(150),
id_prvog NUMBER(4),
id_drugog NUMBER(4),
id_pobednika NUMBER(4),
id_kategorije NUMBER(4) NOT NULL,
CONSTRAINT mecevi_pk PRIMARY KEY(id),
CONSTRAINT mecevi_prvi_fk FOREIGN KEY (id_prvog) REFERENCES TAKMICARI(id),
CONSTRAINT mecevi_drugi_fk FOREIGN KEY (id_drugog) REFERENCES TAKMICARI(id),
CONSTRAINT mecevi_pobednik_fk FOREIGN KEY (id_pobednika) REFERENCES TAKMICARI(id),
CONSTRAINT mecevi_kat_fk FOREIGN KEY (id_kategorije) REFERENCES KATEGORIJE(id)
)
```

Структура креиране табеле може да се мења наредбом ALTER TABLE. Могу да се додају или уклањају колоне, мења тип података колонама, додају или уклањају ограничења, уколико то није у конфликту са већ унетим подацима.

За попуњавање вредности примарног кључа користимо секвенцу, као што је већ описано раније. Секвенца може такође да се креира наредбом упитног језика SQL. Наредбе CREATE TABLE, ALTER TABLE и CREATE SEQUENCE припадају групи DDL наредби.

```
CREATE SEQUENCE MECEVI_SEQ
INCREMENT BY 1
START WITH 1
MAXVALUE 999999
NOCYCLE
CACHE 5
```

Први ред ове наредбе је обавезан. Остали могу да се изоставе. Ова секвенца ће да прави бројеве почев од један, и сваки следећи ће бити за један већи од претходног. Дефинисана је највећа вредност. Након достизања те вредности, ова секвенца неће више генерисати бројеве. У случају да се наведе *CYCLE* уместо *NOCYCLE*, након највеће вредности би почеле да се генеришу вредности из почетка. Може да се дефинише дужина кеша или да се кеш изостави. Када је за секвенцу дефинисан кеш, по неколико вредности из секвенце се припреми у меморији тако да буду спремне за унос у базу. Предност кеша је што доприноси брзини. Мана је што може да се деси да се све вредности из кеша, спремне за унос у базу, изгубе услед неке непредвиђене ситуације (губитак струје и сл.). Наставља се генерисање бројева коришћењем сквенце од места где је стала, па су све вредности из кеша заувек изгубљене. Вредност добијену коришћењем секвенце можемо да употребимо тако што наведемо *NEXTVAL* након назива секвенце, на пример: *MECEVI_SEQ.NEXTVAL*.

Подаци у табеле могу да се уносе наредбом INSERT упитног језика SQL. Следећом наредбом је унет један ред у табелу *Mecevi*. Важно је да се вредности унесу у оном редоследу у којем су колоне у табели, и да буду одговарајућих типова података.

```
INSERT INTO MECEVI
VALUES (MECEVI_SEQ.NEXTVAL, TO_DATE('17/01/2005 09:00:00', 'DD/MM/YYYY HH24:MI:SS'), 1, 'glasanje sudija', 'glasovi 3:2', 21, 10, 21, 7)
```

Можемо, после назива табеле, да наведемо у заградама називе колона. Уколико немамо податак за неку колону, на пример нема другог противника у мечу, пише се *null* за ту колону. Тиме смо експлицитно унели *null* у одговарајуће поље табеле.

```
INSERT INTO MECEVI(id, pocetak, br_kola, tip_rezultata, opis_rezultata, id_prvog, id_drugog, id_pobednika, id_kategorije)
VALUES (MECEVI_SEQ.NEXTVAL, TO_DATE('17/01/2005 09:10:00', 'DD/MM/YYYY HH24:MI:SS'), 1, 'diskvalifikacija', 'nema protivnika', 23, null, 23, 7)
```

Уколико изоставимо називе оних колона за које немамо податак, *null* ће бити имплицитно унет у одговарајуће поље табеле. У следећем примеру недостаје вредност за *opis_rezultata*.

```
INSERT INTO MECEVI(id, pocetak, br_kola, tip_rezultata, id_prvog, id_drugog, id_pobednika, id_kategorije)
VALUES (MECEVI_SEQ.NEXTVAL, TO_DATE('17/01/2005 09:30:00', 'DD/MM/YYYY HH24:MI:SS'), 2, 'poeni', 21, 23, 21, 7)
```

Последња табела која се креира је табела *Poeni*, са примарним кључем *id*, обавезним колонама *vreme* и *bodovi*, опционим колонама *komentar*, *tehnika*, *povrsina* и *kazna*, и два страна кључа, *id_takmicara* и *id_meca*. Када су табеле креиране, уносе се подаци.

Подаци могу да се бришу из табела наредбом DELETE. На пример, следећу наредбу бисмо покренули уколико бисмо желели да обришемо такмичара са шифром 15.

```
DELETE FROM TAKMICARI
WHERE id=15
```

Уколико се изостави услов, биће обрисани сви подаци из табеле, али ће табела остати.

Подаци у редовима табеле се мењају наредбом UPDATE. Следећа наредба ће додати податак о шифри победника у претходно већ унет меч са шифром 3 у табели Mecevi.

```
UPDATE MECEVI
SET id_pobednika=23
WHERE id=3
```

Наредбе INSERT, UPDATE и DELETE припадају групи DML наредби.

## 8. Претраживање базе

Поред могућности складиштења огромних количина података, Oracle систем за управљање базама података омогућава ефикасност у претраживању података и добијању жељених информација из њих. Информације из базе података добијамо упитним блоком SELECT.

Упитни блок се састоји из неколико делова. Обавезни делови су SELECT и FROM, тако да је најкраћи SELECT упит онај којим добијамо све податке из једне табеле. На пример:

```
SELECT * FROM takmicari
```

SELECT упити могу да имају следеће функционалности: селекцију, пројекцију и спајање. Пројекција подразумева избор колона из којих ће се приказати подаци. Списак колона се наводи у SELECT делу SELECT упита. Уколико се наведе звездица (знак *) уместо списка колона, биће приказане све колоне, тј. неће доћи до пројекције. Селекција подразумева издвајање појединих редова. Да би дошло до селекције, потребно је да се основном SELECT упиту дода WHERE део у којем се наводи услов по којем се бирају редови који ће бити приказани. Спајање (енгл. Join) је неопходно када су нам потребни подаци из више од једне табеле.

Следећи упит је пример пројекције:

```
SELECT ime, prezime, adresa, mob FROM takmicari
```

| ID | JMBG | IME | PREZIME | MESTO_RODJ | ADRESA | TELEFON | MOB | TEZINA | NAPOMENA | ID_KLUBA | ID_KATEGORIJE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0605997715123 | Marjana | Majstorovic | Beograd | Majke Jevrosime 33, 11000 Beograd | 011-111-222 | - | 53 | - | 1 | 2 |
| 2 | 3108998715234 | Milica | Zoranovic | Beograd | Francuska 12, 11000 Beograd | 011-222-333 | 062-454-121 | 48 | - | 1 | 2 |
| 3 | 3006997715321 | Ana | Milosavljevic | Smederevo | Kolarceva 15, 11000 Beograd | 011-333-444 | 064-212-323 | 53 | - | 2 | 2 |
| 4 | 1207998715110 | Lena | Kulic | Beograd | Kosovska 10, 11000 Beograd | 011-444-555 | - | 49 | - | 1 | 2 |
| 5 | 1002998715201 | Olivera | Stosic | Beograd | Vasina 22, 11000 Beograd | 011-555-777 | 061-989-676 | 50 | - | 4 | 2 |
| 6 | 3009997715007 | Tamara | Gavrilovic | Smederevo | Svetogorska 45, 11000 Beograd | 011-777-888 | - | 50 | - | 3 | 2 |
| 8 | 0605998715006 | Slavica | Mastilovic | Beograd | Takovska 5, 11000 Beograd | 011-888-555 | - | 51 | - | 4 | 2 |
| 9 | 1205997715550 | Lidija | Antunovic | Beograd | Cara Dusana 128/1, 11000 Beograd | 011-999-111 | - | 54 | - | 3 | 2 |
| 10 | 1405998710055 | Pavle | Milosevic | Beograd | Pozeska 45, 11000 Beograd | 011-468-135 | - | 70 | - | 4 | 7 |
| 21 | 2006998710550 | Branislav | Zoranovic | Beograd | Kneza Milosa 124, 11000 Beograd | 011-123-987 | 011-377-992 | 69 | - | 1 | 7 |

Слика 65. Пројекција

Селекцијом добијамо избор редова. Следећи пример упита садржи селекцију и приказују се само они такмичари који тренирају у клубу са шифром 1.

```
SELECT * FROM takmicari WHERE id_kluba=1
```

| ID | JMBG | IME | PREZIME | MESTO_RODJ | ADRESA | TELEFON | MOB | TEZINA | NAPOMENA | ID_KLUBA | ID_KATEGORIJE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0605997715123 | Marjana | Majstorovic | Beograd | Majke Jevrosime 33, 11000 Beograd | 011-111-222 | - | 53 | - | 1 | 2 |
| 2 | 3108998715234 | Milica | Zoranovic | Beograd | Francuska 12, 11000 Beograd | 011-222-333 | 062-454-121 | 48 | - | 1 | 2 |
| 3 | 3006997715321 | Ana | Milosavljevic | Smederevo | Kolarceva 15, 11000 Beograd | 011-333-444 | 064-212-323 | 53 | - | 2 | 2 |
| 4 | 1207998715110 | Lena | Kulic | Beograd | Kosovska 10, 11000 Beograd | 011-444-555 | - | 49 | - | 1 | 2 |
| 5 | 1002998715201 | Olivera | Stosic | Beograd | Vasina 22, 11000 Beograd | 011-555-777 | 061-989-676 | 50 | - | 4 | 2 |
| 6 | 3009997715007 | Tamara | Gavrilovic | Smederevo | Svetogorska 45, 11000 Beograd | 011-777-888 | - | 50 | - | 3 | 2 |
| 8 | 0605998715006 | Slavica | Mastilovic | Beograd | Takovska 5, 11000 Beograd | 011-888-555 | - | 51 | - | 4 | 2 |
| 9 | 1205997715550 | Lidija | Antunovic | Beograd | Cara Dusana 128/1, 11000 Beograd | 011-999-111 | - | 54 | - | 3 | 2 |
| 10 | 1405998710055 | Pavle | Milosevic | Beograd | Pozeska 45, 11000 Beograd | 011-468-135 | - | 70 | - | 4 | 7 |
| 21 | 2006998710550 | Branislav | Zoranovic | Beograd | Kneza Milosa 124, 11000 Beograd | 011-123-987 | 011-377-992 | 69 | - | 1 | 7 |

Слика 66. Селекција

Уз остале податке о такмичарима у табели *Takmicari* имамо само шифру клуба у којем тренирају. Уколико желимо да видимо за сваког такмичара и све податке о његовом клубу, потребно је да спојимо табеле *Takmicari* и *Klubovi* у FROM делу упита. У ON делу упита се наводи услов по којем ће се извршити спајање, а то је најчешће једнакост вредности страног кључа из једне табеле и примарног кључа из друге. Свака колона припада некој табели, па испред назива колоне може, а у ситуацијама када имамо колоне са истим називима у различитим табелама мора, да се дода име табеле.

```
SELECT * FROM takmicari JOIN klubovi ON (takmicari.id_kluba=klubovi.id)
```

Највећи број упита има и селекцију и пројекцију и спајање. Следећим упитом се уз име и презиме такмичара, приказује и назив клуба у којем тренира и то само за оне такмичаре који се такмиче у категорији са шифром 2. Неколико различитих података могу да се споје у једну колону оператором спајања са ознаком ||. Уколико се не наведе другачије, називи колона у резултату одговарају изразима и називима колона из табела. Колонама могу да се доделе и другачији називи. Уколико додељени назив има више од једне речи или нам је важно да буду приказана велика и мала слова, назив стављамо између наводника. Кључна реч AS којом наглашавамо да смо некој колони доделили нови назив приликом приказа резултата, је опциона и може да се изостави.

```
SELECT ime||' '||prezime AS "Takmicar", naziv AS "Naziv kluba"
FROM takmicari JOIN klubovi ON (takmicari.id_kluba=klubovi.id)
WHERE id_kategorije=2
```

Резултат је приказан у виду табеле.

Results Explain Describe Saved

| Takmicar | Naziv Kluba |
|---|---|
| Marjana Majstorovic | Cukarica |
| Milica Zoranovic | Cukarica |
| Ana Milosavljevic | Nippon |
| Lena Kulic | Cukarica |
| Olivera Stosic | Roda |
| Tamara Gavrilovic | Sunce |
| Slavica Mastilovic | Roda |
| Lidija Antunovic | Sunce |

8 rows returned in 0,03 seconds

Слика 67. Списак такмичара и њихових клубова

Исти упит може да се преправи тако да се при покретању захтева унос шифре категорије за коју нас интересује списак такмичара. Параметар се наводи након двотачке. Поред тога, резултат може и да се уреди алфабетно по називу клуба, а уколико двоје тренира у истом клубу, по презимену и имену такмичара. Услови сортирања се наводе у ORDER BY делу упита.

```
SELECT ime||' '||prezime AS "Takmicar", naziv AS "Naziv kluba"
FROM takmicari JOIN klubovi ON (takmicari.id_kluba=klubovi.id)
WHERE id_kategorije=:kategorija
ORDER BY naziv, prezime, ime
```

Табела *Poeni* има податке о сваком освојеном поену. У табели постоје страни кључеви, шифра такмичара који је остварио поен и шифра меча у којем је дошло до поена. У табли мечеви постоји шифра категорије којој припада меч. Назив категорије је у табли *Kategorije*. Уколико желимо извештај са подацима о свим освојеним поенима у првом колу турнира, укључујући назив категорије, време почетка меча, име такмичара који је освојио поен и детаље о освојеном поену, потребно је да спојимо четири табеле: *Poeni*, *Mecevi, Takmicari* и *Kategorije*. У FROM делу се споје прво две табеле, па се изврши спајање са трећом, па са четвртом. Резултат можемо да сортирамо по категорији. У оквиру исте категорије можемо да сортирамо по почетку меча, а затим по секунди у којој је дошло до поена.

```
SELECT kategorije.naziv AS "Kategorija", TO_CHAR(pocetak, 'HH24:MI:SS') AS "Pocetak meca", prezime AS "Takmicar",
vreme AS "Sekunda", tehnika||' '||povrsina AS "Tehnika", kazna AS "Kazna", bodovi AS "Broj poena"
FROM takmicari JOIN poeni ON (takmicari.id=poeni.id_takm) JOIN mecevi ON (mecevi.id=poeni.id_meca)
JOIN kategorije ON (mecevi.id_kategorije=kategorije.id)
WHERE br_kola=1
ORDER BY "Kategorija", "Pocetak meca", "Sekunda"
```

| Kategorija | Pocetak Meca | Takmicar | Sekunda | Tehnika | Kazna | Broj Poena |
|---|---|---|---|---|---|---|
| Kadetkinje - srednja | 09:40:00 | Gavrilovic | 31 | ruka telo | - | 1 |
| Kadetkinje - srednja | 09:40:00 | Mastilovic | 93 | vezane tehnike | - | 2 |
| Kadetkinje - srednja | 09:50:00 | Stosic | 19 | ruka telo | - | 1 |
| Kadetkinje - srednja | 09:50:00 | Stosic | 40 | | drugi izlazak protivnika iz borilista | 1 |
| Kadetkinje - srednja | 09:50:00 | Stosic | 68 | | treci izlazak protivnika iz borilista | 2 |
| Kadetkinje - srednja | 10:10:00 | Zoranovic | 24 | noga glava | - | 3 |
| Kadetkinje - srednja | 10:10:00 | Majstorovic | 56 | ruka telo | - | 1 |
| Kadetkinje - srednja | 10:10:00 | Zoranovic | 82 | vezane tehnike | - | 2 |

Слика 68. Сви освојени поени у првом колу турнира

Постоје многе функције које могу да се користе у упитима. Неке од њих су функције конверзије које преводе податак из једног типа података у други. Таква је, на пример, функција TO_CHAR. Све функције се деле на оне које враћају резултат за сваки ред у табели и оне које враћају резултат за групу редова у табели. У следећем примеру имамо употребу две функције које враћају резултат за сваки ред. Функција UPPER пребацује текст у велика слова, и приказаће свако презиме из табеле *Takmicari* великим словима. Функција SUBSTR издваја део текста од наведене позиције и наведене дужине. Како су први карактери у податку ЈМБГ дан и месец рођења (по два знака), рођендан сваког такмичара можемо да извучемо из тог податка.

```
SELECT UPPER(Prezime) "Takmicar", SUBSTR(jmbg,1,2)||'.'||SUBSTR(jmbg,3,2)||'.' AS "Rodjendan"
FROM takmicari
```

Уколико желимо да добијемо укупан број одржаних мечева потребна нам је функција која враћа један резултат за целу табелу.

```
SELECT COUNT(id) AS "Ukupan broj meceva" FROM MECEVI
```

Могуће је добити и број мечева за сваку категорију. Потребно је груписати редове према категорији и онда извршити функцију која пребројава.

```
SELECT id_kategorije "Kategorija", COUNT(id) AS "Broj meceva"
```

```
FROM mecevi GROUP BY id_kategorije
```

Наравно, уколико бисмо желели и назив категорије, упит би морали да проширимо спајањем са табелом *Kategorije*.

```
SELECT kategorije.naziv "Kategorija", COUNT(mecevi.id) AS "Broj meceva"
FROM mecevi JOIN kategorije ON (mecevi.id_kategorije=kategorije.id)
GROUP BY kategorije.naziv
```

| Kategorija | Broj Meceva |
|---|---|
| Kadeti - teska | 3 |
| Kadetkinje - srednja | 7 |

Слика 69. Укупан број мечева за сваку категорију

Следећим упитом добијамо победника за категорију са шифром 2. Табеле *Takmicari* и *Mecevi* се спајају тако да се сваком мечу дода његов победник. Изврши се селекција оних мечева који припадају жељеној категорији. Победник у целој категорији је онај који је победио у последњем мечу, тј. у оном мечу код којег је број кола највећи могући у категорији са шифром 2.

```
SELECT ime||' '||prezime "Pobednik"
FROM takmicari JOIN mecevi ON (mecevi.id_pobednika=takmicari.id)
WHERE mecevi.id_kategorije=2 AND
br_kola=(SELECT MAX(br_kola) FROM mecevi WHERE id_kategorije=2)
```

Нешто је тежи упит којим добијамо победника за сваку категорију.

```
SELECT k.naziv "Kategorija", ime||' '||prezime "Pobednik"
FROM takmicari t JOIN mecevi m ON (m.id_pobednika=t.id)
JOIN kategorije k ON (m.id_kategorije=k.id)
WHERE br_kola=(SELECT MAX(br_kola) FROM mecevi m1 WHERE m1.id_kategorije=m.id_kategorije)
```

Можемо добити и просечан број освојених поена у свим одржаним мечевима. Ово подразумева да се прво израчуна сума освојених бодова за сваки појединачни меч употребом функције SUM, а затим да се израчуна просек свих тих вредности употребом функције AVG.

```
SELECT AVG(SUM(bodovi)) FROM poeni GROUP BY id_meca
```

Као што је већ речено раније, програмски језик PL/SQL је проширење упитног језика SQL и има могућност дефинисања променљивих, као и управљања током програма. У овом програмском језику могу да се пишу неименовани и именовани блокови. Именовани блокови су функције и процедуре које се меморишу у систему и могу да се позивају по потреби. Овде ће бити приказан један неименовани блок којим се приказује детаљни извештај са такмичења који покрива списак свих категорија, а за сваку категорију, списак свих мечева и освојених поена у њима.

Неименовани PL/SQL блок се састоји из два дела, одељка за декларацију променљивих и извршног блока. Одељак за декларацију променљивих почиње резервисаном речју DECLARE. Наредбе блока се пишу између речи BEGIN и END. Променљиве можемо да користимо да бисмо у њих превукли податке из базе ради даље обраде и приказа. За сваку променљиву морамо да дефинишемо тип података. То може да буде неки од основних типова података, на пример NUMBER, или да се променљивој додели исти тип као тип података дефинисан за неку колону у

бази. У том случају, потребно је да се наведе реч TYPE после назива одговарајуће табеле и колоне, на пример mecevi.id%TYPE.

Курсор (енгл. Cursor) је још једна врста објекта у бази података. Служи да омогући појединачни приступ сваком реду у табели. Дефинише се у одељку за декларацију. У извршном блоку, курсор мора да се отвори наредбом OPEN и након употребе да се затвори наредбом CLOSE. Како углавном свака табела има више редова, читање једног реда се стави унутар циклуса LOOP који се понавља све док курсор не стигне до краја табеле, односно док се не дође до ситуације да курсор не може да нађе више података, тј. naziv_kursora%NOTFOUND. Помоћу функција PUT_LINE из пакета DBMS_OUTPUT приказује се резултат.

Следи неименовани PL/SQL блок којим се приказује списак свих мечева и освојених поена у њима за све категорије.

```
DECLARE
  CURSOR c_kat IS SELECT id, naziv FROM kategorije;

  CURSOR c_mec (p_kat NUMBER) IS
  SELECT m.id, m.pocetak, t1.prezime, k1.naziv, t2.prezime, k2.naziv, t.prezime, m.tip_rezultata
  FROM mecevi m JOIN takmicari t1 ON (m.id_prvog=t1.id)
  JOIN klubovi k1 ON (t1.id_kluba=k1.id)
  JOIN takmicari t2 ON (m.id_drugog=t2.id)
  JOIN klubovi k2 ON (t2.id_kluba=k2.id)
  JOIN takmicari t ON (m.id_pobednika=t.id)
  WHERE id_kategorije=p_kat
  ORDER BY m.pocetak;

  CURSOR c_poen (p_mec NUMBER) IS
  SELECT p.vreme, t.prezime, p.bodovi
  FROM poeni p JOIN takmicari t
  ON (p.id_takm=t.id)
  WHERE p.id_meca=p_mec;

  v_kat_id kategorije.id%TYPE;
  v_kat_naziv kategorije.naziv%TYPE;
  v_mec mecevi.id%TYPE;
  v_poc mecevi.pocetak%TYPE;
  v_prvi takmicari.prezime%TYPE;
  v_drugi takmicari.prezime%TYPE;
  v_pobednik takmicari.prezime%TYPE;
  v_klub1 klubovi.naziv%TYPE;
  v_klub2 klubovi.naziv%TYPE;
  v_rez mecevi.tip_rezultata%TYPE;
  v_vreme poeni.vreme%TYPE;
  v_poentirao takmicari.prezime%TYPE;
  v_bod poeni.bodovi%TYPE;
  i NUMBER :=0;

BEGIN
  OPEN c_kat;
  LOOP
    FETCH c_kat INTO v_kat_id, v_kat_naziv;
    EXIT WHEN c_kat%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE('KATEGORIJA: '||UPPER(v_kat_naziv));
```

```
    DBMS_OUTPUT.PUT_LINE(' ');
    OPEN c_mec(v_kat_id);
    i:=0;
    LOOP
      FETCH c_mec INTO v_mec, v_poc, v_prvi, v_klub1, v_drugi, v_klub2, v_pobednik, v_rez;
      EXIT WHEN c_mec%NOTFOUND;
      i:=i+1;
      DBMS_OUTPUT.PUT_LINE('  MEC BR. '||i);
      DBMS_OUTPUT.PUT_LINE('  POCETAK '||TO_CHAR(v_poc, 'HH24:MI:SS'));
      DBMS_OUTPUT.PUT_LINE('  '||v_prvi||' ('||v_klub1||') - '||v_drugi||' ('||v_klub2||')');
      OPEN c_poen(v_mec);
      LOOP
        FETCH c_poen INTO v_vreme, v_poentirao, v_bod;
        EXIT WHEN c_poen%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE('    '||v_vreme||'s; '||v_poentirao||'; '||v_bod||' poena');
      END LOOP;
      CLOSE c_poen;
      DBMS_OUTPUT.PUT_LINE('    POBEDNIK: '||v_pobednik||'; '||v_rez);
      DBMS_OUTPUT.PUT_LINE(' ');
    END LOOP;
    CLOSE c_mec;
    DBMS_OUTPUT.PUT_LINE(' ');
  END LOOP;
  CLOSE c_kat;
END;
```

Део резултата овог неименованог PL/SQL блока је приказан на следећој слици.

```
KATEGORIJA: KADETKINJE - SREDNJA

  MEC BR. 1
  POCETAK 09:40:00
  Mastilovic (Roda) - Gavrilovic (Sunce)
    31s; Gavrilovic; 1 poena
    93s; Mastilovic; 2 poena
    POBEDNIK: Mastilovic; poeni

  MEC BR. 2
  POCETAK 09:50:00
  Stosic (Roda) - Kulic (Cukarica)
    19s; Stosic; 1 poena
    40s; Stosic; 1 poena
    68s; Stosic; 2 poena
    POBEDNIK: Stosic; diskvalifikacija

  MEC BR. 3
  POCETAK 10:00:00
  Milosavljevic (Nippon) - Antunovic (Sunce)
    POBEDNIK: Milosavljevic; glasanje sudija

  MEC BR. 4
  POCETAK 10:10:00
  Majstorovic (Cukarica) - Zoranovic (Cukarica)
    24s; Zoranovic; 3 poena
    56s; Majstorovic; 1 poena
    82s; Zoranovic; 2 poena
    POBEDNIK: Zoranovic; poeni
```

Слика 70. Списак свих мечева и освојених поена у њима за све категорије

# 9. Закључак

Као један од система за управљање базама података, *Oracle Database 10g* омогућава складиштење података, као и ефикасну претрагу и добијање корисних информација. Најважније за добру и функционалну базу података је добар пројекат на основу којег ће база бити изграђена. Дијаграм ентитета и веза (ER дијаграм) приказује све потребне податке, као и везе између њих, и служи као основа за изградњу базе података. Сваки дијаграм мора да има ентитете и везе. У овом раду описано је како се може моделирати турниријско такмичење помоћу Oracle софтверских алата уз коришћење 27 ентитета.

Ентитети имају атрибуте. На пример, за турнирско такмичење су важни такмичари, па постоји ентитет *Takmicar*, са атрибутима који га описују, као што су: *ime*, *prezime*, *JMBG*... Поред тога, постоје и ентитети *Klub, Trener, Mec, Poen, Sudija, Boriliste*... За сваки енитет са дијаграма се креира табела, која има оне колоне које одговарају атрибутима на пројекту. За ентитет *Takmicar* ће бити креирана табела *Takmicari* са колонама: *ime*, *prezime*, *jmbg*...

Ентитети су повезани. На пример: такмичар тренира у клубу, на борилишту се одржавају мечеви, судија суди у мечу, такмичари освајају поене у мечевима... Свака веза има назив, кардиналност и опционалност. Везе са пројекта у бази података постају страни кључеви, додатне колоне које чувају вредности кључева из других табела и служе за повезивање података. Тако, на пример, у табели *Takmicari* имамо страни кључ, шифру клуба у којем такмичар тренира, док детаље о клубу чувамо у табели *Klubovi*.

Поред ентитета и веза, ER дијаграм може да садржи и друге структуре, као што су хијерархијска и рекурзивна структура, под и над типови, лукови и генерички модел. Сви они служе за пројектовање неких специфичних ситуација. Сваку ситуацију можемо да моделирамо помоћу ових дијаграма на више начина. Важно је изабрати најбољи могући. Један од алата који омогућава креирање ER дијаграма је *Oracle Data Modeler*.

У овом раду је приказан и један мањи упрошћен модел са 5 ентитета на основу којег је креирана база података. Тиме су приказане све фазе потребне за припрему и процес креирања базе која онда може да чува податке и из које могу да се добијају корисне информације.

Сама база података је често основа информационог система који подразумева и неку апликацију са приступачним кориснички интерфејсом, и у раду са којом крајњи корисници приступају подацима, уносе их и мењају и добијају информације. Само илустративно су показани неки упити упитног језика SQL и неименовани PL/SQL блок помоћу којих можемо да добијемо информације из базе. Могућности ових језика су много веће и нису централна тема овог рада.

## Литература

[1] Сајт Oracle Академије
https://academy.oracle.com/

[2] Сајт компаније Oracle
http://www.oracle.com

[3] Сајт Сведске карате федерације
http://www.wkf.net/images/stories/downloads/wkf_organisation_rules_2011.pdf

[4] Сајт Београдског карате савеза – http://www.beokarate.rs/bks/registracija%2007-08/obrasci.pdf

[5] http://www.infogo.biz/pib-poreski-identifikacioni-broj.html

[6] Трофеј Чукарице, пријава на турнир
http://www.karatecukarica.rs/Trofej%20Cukarice%202011.pdf

[7] Сајтови карате клубова
www.karatecukarica.rs
www.nippon.org.rs
www.srksunce.rs

[8] http://karate-pravila.bloger.hr/