Факултет инжењерских наука

Универзитет у Крагујевцу

Петар Р. Љушић

Тема: Моделирање база података у програмском пакету „Visio“

-завршни рад-

Крагујевац, 2017.

Универзитет у Крагујевцу

Факултет инжењерских наука

Назив студијског програма: Машинско инжењерство

Ниво студија: Основне академске студије

Модул: Информатика у инжењерству

Предмет: Базе података

Број индекса: 72/2013

Петар Р. Љушић

Моделирање база података у програмском пакету „Visio“

- завршни рад-

Комисија за преглед и одбрану:

1. Проф. др Милан Ерић **-** ментор
2. _____
3. _____

Датум одбране:_____

Оцена:_____

У оквиру завршног рада са темом "*Моделирање база података у програмском пакету "Visio"***" кандидат треба да:**

Проучи литературу и презентира основе коришћења програмског пакета висио за моделирање база података. Завршни рад треба да обухвати уводна разматрања везана за моделирање база података, карактеристике програмског пакета висио, подешавање окружења за моделирање база података и конкретан пример употребе. На крају рада дати закључна разматрања.

Препоручена литература:

1. Лазаревић Б. …..: ***Базе података***, ФОН Београд, Београд 2003.
2. Павловић-Лажетић Г.: ***Основе релационих база података***, Математички факултет, Београд, 2000.
3. Bonnie Biafore: ***Visio 2007 Bible***, Wiley Publishing, Inc., Indianapolis, Indiana, 2007.

Крагујевац, октобар 2016.

Ментор:

Prof. Dr Milan D. Erić

_____

Изјава о самосталној изради рада

Изјављујем да сам овај завршни рад урадио самостално, служећи се стеченим знањем и искуством као и наведеном литературом.

72/2013 Петар Љушић

_____

(потпис)

Резиме

У овом завршном раду биће представљен Microsoft Visio 2010 програмски пакет у функцији пројектовања и употребе базе података. Такође ћу објаснити врсте и потребе за базом података као и њен начин функционисања који ће бити приказан на конкретном примеру података.

Кључне речи: База података, Visio 2010, ER модел

Summary

In this paper will be represented Microsoft Visio 2010 software package as a function of the design and way to use databases. I will also explain the types and the necessity for a database as well its mode of functioning which will be shown on a specific example.

Keywords: Database, Visio 2010, ER model

Садржај

1. Увод

Од свих проналазака у двадесетом веку аутомобил и рачунар су најчешће у употреби. Рачунари су престали да буду алатка коју само користе високостручни људи, превазишли су своју првобитну намену и готово да нема области у којој нису нашли примену. Развојом рачунарства, имплементацијом у образовање и привреду, рачунари су постали део свакодневнице. Потреба за смену писаних-штампаних спискова, докумената, података и сл. је постала могућа, тако да су недостаци попут лаког губљења података, немогућности истовременог приступа, немогућности ажурирања, претраживања али и много других недостатака полако одлазили у заборав. Дакле, развојом рачунара, настале су и прве базе података и увеле су револуцију у коришћењу реалних података и рачунара. Највећи недостатак рачунара сем хардверских недостатака је био и интернет (нпр. брзина протока података за умрежавања банака и разних других установа ради прављења велике „online“ базе података), што је условило, с једне стране, велика улагања у развој мрежне инфраструктуре. База података је колекција података организованих за брзо претраживање и приступ, која заједно са системом за администрацију, организовање и меморисање тих података, чини систем базе података. Из угла корисника, подаци су на неки логички начин повезани. Корисници приступају бази података првенствено преко упита, коришћењем кључних речи или неких софтвера, корисници могу брзо да пронађу, преуреде, групишу и одаберу област која им рецимо треба за састављање извештаје о скупу података. Базу података, по правилу, користи велики број корисника. Данас, велике компаније које се баве интернет сервисима зарађују огроман новац, нудећи тржишту опцију названом „Cloud“ , која представља складиштење података на сервере компанија које се налазе на удаљеној локацији. Пребацивање фотографија, докумената или базе података као и сам приступ се врши преко интернета. На тај начин се могу правити и резервне копије база, како би се осигурале од губљења података. Ово све доказује значај база података у данашњем свету.

Слика 1. „Google cloud“ лого

Слика 2. „Google date“ центар у Финској

2. База података

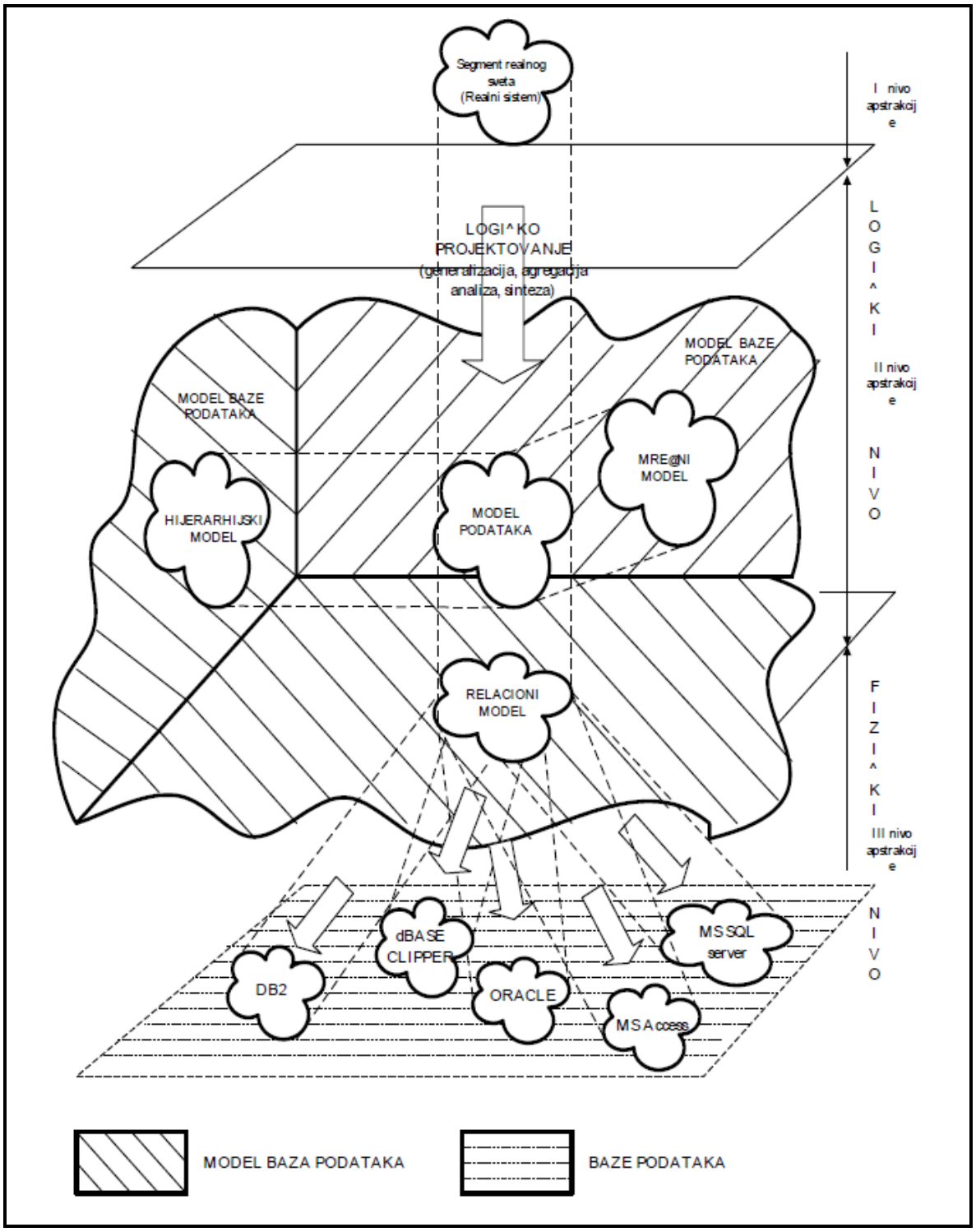
Смисао рачунарске базе података се треба огледати у што објективнијој слици реалног система. Базу података треба најпре пажљиво испланирати, па тек онда креирати и обликовати. База података представља основу сваког информационог система. Базе података могу да складиште информације о особама, производима, поруџбинама или било чему другом. Многе базе започињу свој развој као листа у неком текст едитору или у табели, а како листа расте, почињу да се појављују вишкови и недоследности у подацима. Подаци постају тешки за разумевање у облику листе и ограничени су начини прегледања или претраживања. Када проблеми почну да се јављају, потребно је пребацити податке у базу података који је направио систем за управљање базом података (DBMS-Datebase menager system).

2.1 Модели података

Систем база података се може посматрати као хијерархија апстракција. Сваки ниво у тој хијерархији апстракција је једна врста модела и представља скуп ентитета односно објекта, веза и операција између њих. Задатак пројектовања је да рачунарски обезбеди најнижи ниво потребне апстракције.

Графички приказ хијерархија апстракција се може видети на слици 3., први ниво (гледајући одозго на доле) чини ниво апстракције који се темељи на сегменту реалног света, информациони задатак или потребу. Следећи ниво те хијерархије је ниво модела података или логичка слика података. На следећем нивоу налази се репрезентација модела погодних структура података, на овом нивоу не постоје никакви утицаји особина рачунарског система на модел података. За разлику од предходних нивоа на најнижем нивоу хијерархије апстракција, су дозвољени утицаји особина рачунарског система на модел података, овај ниво се назива физички ниво и ту су подаци представљени у бинарном облику са свим специфичним карактеристикама.

Моделирање или логичко пројектовање, бави се проблемима адекватне употребе модела у представљању ентитета и односа (веза) између њих на основу анализе информационих захтева и информационог окружења. Значај доброг логичког пројектовања базе података лежи у чињеници да су са таквом базом манипулисање, одржавање, ажурирање и све остале потребне операције над тим подацима најефикасније [1]. За моделирање се користе две методе , у зависности од модела треба одабрати одговарајући метод. Метода анализе, којој одговара принцип пројектовања од врха на доле, подразумева анализу улаза и излаза како би се из њих изводили атрибути, ентитети или синтеза групе атрибута и ентитета. Методи синтезе одовара обрнути принцип пројектовања.



Слика 3. Графички приказ пројектовања база података

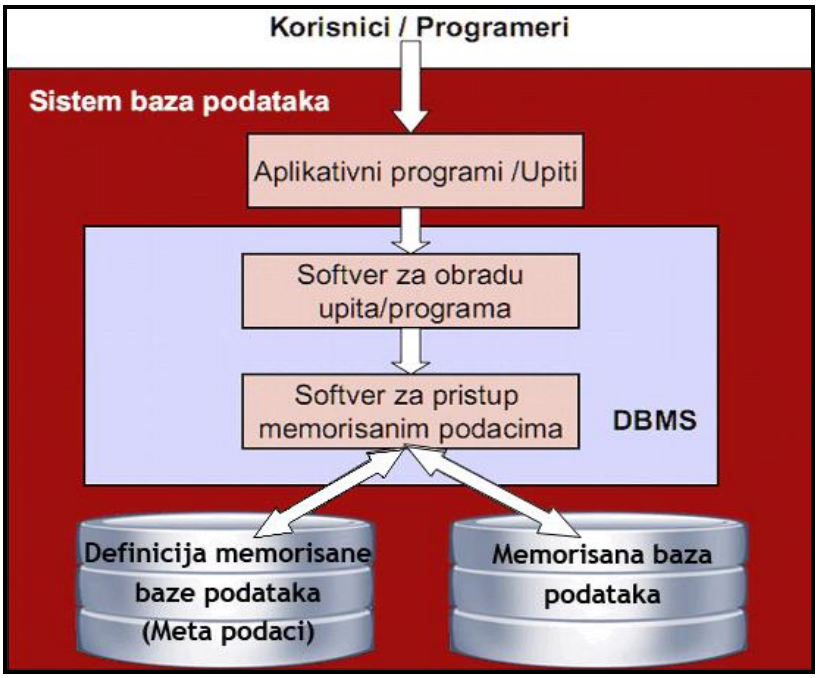
2.2 Систем за управљање базом података (Datebase menager system)

Систем за управљање базом података је софтверски систем за ефикасан унос, чување, претраживање, контролу приступа и одржавање великог броја података. Једна база података може да садржи више табела, рецимо одређени систем који користи три табеле не представља три базе, већ једну која садржи три табеле.

Осим ако база података није дизајнирана тако да користи податке или код из другог извора. Систем за управљање базом података омогућава постизање следећих циљева:

1. трајно чување података, независност од процеса који те податке користе, ефикасан приступ.
2. креирање нових база података као и њихових логичких структура података помоћу језика за дефинисање података (data-definition language)
3. лакша измена и приступ подацима захваљујући упитним језицима (query language или data-manipulation language)
4. минималност дуплирања података (редундансе)
5. могућност опоравка након грешке у систему
6. могућност приступа подацима омогућеним корисницима
7. заштита података која се огледа у очувању интегритета базе (тачност, коректност података) и сигурности података (од неовлашћеног приступа и могућност ажурирања података)

За постизање ових циљева поред система за управљање мора се детаљно сагледати потреба корисника и формирати најближи логички односно релациони модел. На слици 4. можемо видети интегрисани приказ базе података и околине.



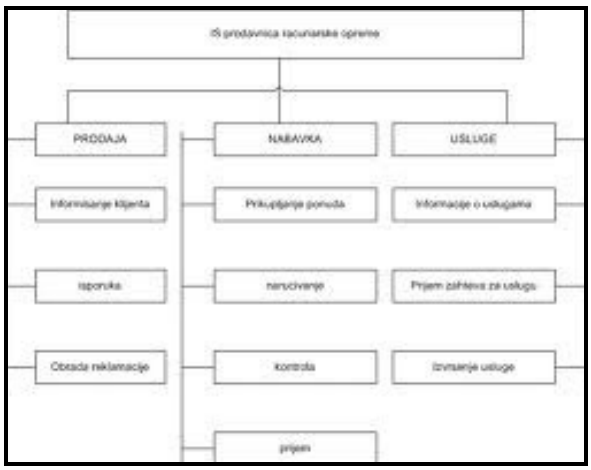
Слика 4. Графички приказ околине система базе података

Код „DBMS“ (Datebase menager system) разликујемо следеће моделе:

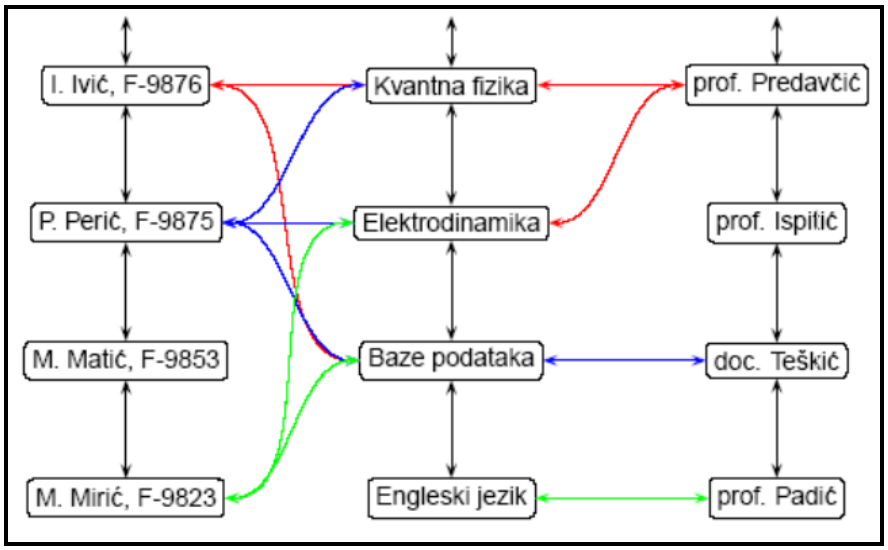
1. Хијерархијски модел
2. Мрежни модел
3. Релациони модел
4. Модел ентитети-везе (ER, MOV)
5. Semantic data model (SDM)
6. Објектно-орјентисани модел и др.

Хијерархијски модел базе података појавио се педесетих година прошлог века. Можда нам на први поглед хијерархијски приступ проблему (Слика 5.) изгледа и делује као прихватљив, разумљив и једноставан, али потребно је нагласити да је са становишта оптималног пројектовања и манипулације подацима, реч о неповољном моделу. Лоша страна хијерархијског модела је у првом реду недостатак егзактне математичке теорије која би омогућила његову пуну имплементацију на рачунару.

Мрежни модел информационог система настао је као проширење хијерархијског. Основна разлика у односу на хијерархијски модел, лежи у чињеници да се омогућило да "наследник" има више "родитеља", што значи да овакав модел прихвата и везе типа M:N. На тај начин везе међу слоговима појединих табела, графички интерпретиране, личе на паукову мрежу (Слика 6.) по чему је модел и добио име.



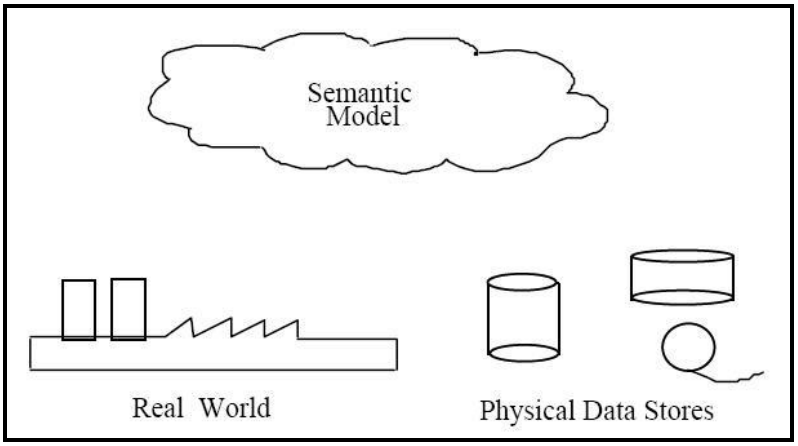
Слика 5. Графички приказ Хијерархијског модела



Слика 6. Графички приказ мрежног модела

Semantic data model (SDM) Слика 7. подразумева коришћење техника за дефинисање значења података у контексту својих веза с другим подацима. Ова, наизглед збуњујућа, реченица добро сажима о чему се заправо ради у семантичком

моделовању базе података. Није погрешно рећи како се семантичко моделирање базе података одвија у исто време кад и прикупљање података потребних за изградњу базе података, јер док прикупљамо податке морамо знати која је њихова улога, односно њихово значење, у стварном свету, како би могли знати која је њихова улога у будућој бази. Подаци су међусобно повезани, било директно или индиректно, те се међусобно допуњавају.

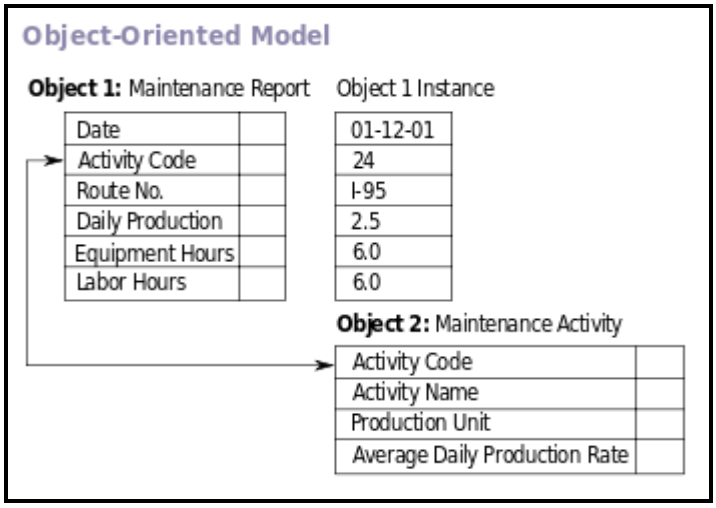


Слика 7. Графички приказ „Semantic data“ модела

Објектно-орјентисани модел (Слика 8.), произвођачи релационих система су проширили своје производе новом функционалношћу, као што је:

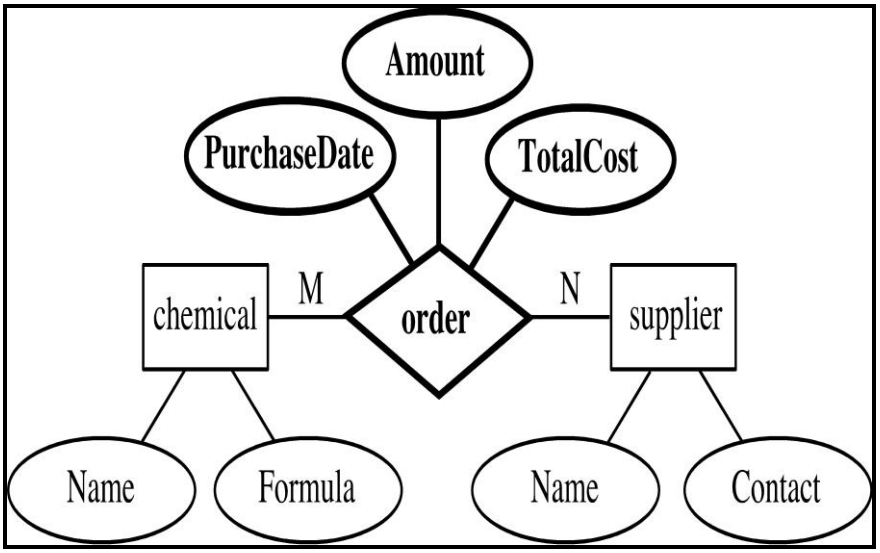
1. подршка богатијем скупу типова и правила,
2. наслеђивање атрибута у односима типа генерализација/специјализација,
3. учаурење(encapsulation) података и операција (функција, процедура), при чему је начин приступања подацима ограничен дефинисаним операцијама (функцијама, процедурама).

Ова проширења представљају саставни део објектно орјентисаних система, па се зато и увео нови правац у развоју база података-објектно орјентисаног модела, и имплементација система за управљање базама података над тим моделом.



Слика 8. Графички приказ Објектно-орјентисаног модела

Модел ентитети-везе (објекти-везе MOV, ER (entity-relationship) model, приказан на Слици 9.) је концептуални модел база података који даје поглед на скуп података и на везе које их карактеришу. Израда овог модела је сложен задатак, али то је задатак које је од суштинског значаја за изградњу велике, трајне и издржљиве базе података. Модел ентитет-везе се ослања на три базна коцепта: својство, ентитет и везу. Својство је најмањи податак који описује један ентитет или везу. Ентитет је апстрактни појам којим се описује скуп сличних објеката, а веза је апстрактни појам којом се описује тип везе између два или више ентитета.



Слика 9. Графички приказ модела Ентитет-веза

Релациони модел је данас најпопуларнији модел база података и то захваљујући пре свега следећим особинама:

1. структура модела је веома једноставна,
2. база података представља скуп табела, и
3. могућа је формално-математичка интерпретација табела.

Као што му и само име говори, овај модел се заснива на табелама и релацијама међу њима. Како се и релација (веза) између табела може у релационом моделу опет приказати као табела, у релационом моделу може се све дефинисати табелама. Захваљујући добро развијеном математичком апарату релационе алгебре лако се имплементира на рачунару. Код релационих база података разликујемо базне и изведене релације.

У овом раду ћу детаљније говорити о документовању и пројектовању логичких модела база података за релационе базе података.

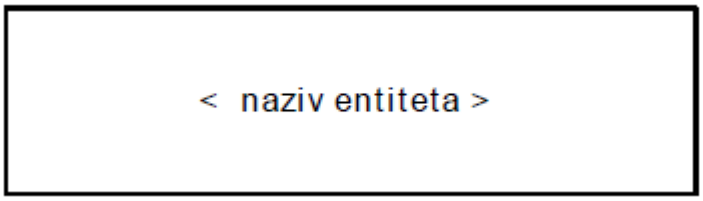
2.3 ER (Entity Relationship) модел података

ER модел података је концептуални модел који се користи за приказивање верног одраза базе података засноване на правилима релационог модела, спада у методу логичког пројектовања. Ово је само помоћна фаза моделовања (међу корак) из које се касније лако дефинише база података. Овај модел је битан због реализације појма ентитета, атрибута и везе који су кључни елементи свих база података. Основно својство овог модела је дијаграмска техника којом се објекти података представљају визуално.

Ентитет подразумева објекат посматрања који се може издвојити из околине и описати (једнозначно одређивање неког елемента-објекта). Ентитет може бити готово било шта, нпр. :

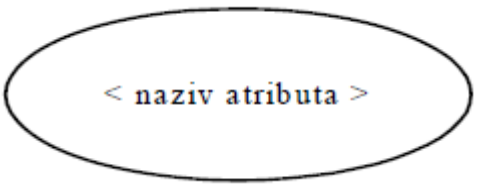
1.реални објекат (особа , машина, кућа, документ…)
2.апстрактни појам (мера, количина, боја, радно место, итд.)
3.догађај (рођење, упис, исплата, прекршај…)

Ентитет се графички приказује правоугаоником у оквиру кога се уписује назив типа ентитета у једнини (Слика 10.)



Слика 10. Графички приказ ентитета

Атрибут ближе одређује ентитет, сваки ентитет има различита својства (обележја) атрибута, свако то обележје има своје име и вредност. Скуп атрибута који описују ентитет и име ентитета одређује тип ентитета. Може постојати већи број примерака (појава) ентитета задатог типа (на пример КЊИГА је тип чији су примерци „Мали човек“, „Петар Пан“, итд.). Графички приказ атрибута је приказан на слици Слика 11.



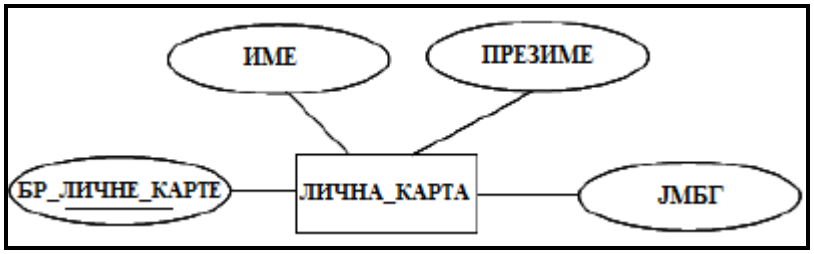
Слика 11. Графички приказ атрибута

Кључ, кандидат за кључ је атрибут, или скуп атрибута, чије вредности једнозначно одређују примерак ентитета задатог типа, односно јединствени идентификатор у фази имплементације постаје примарни кључ. Атрибути који једнозначно идентификују појаву ентитета симболички означавамо знаком #, или

подвлачењем атрибута. Дакле, не могу постојати два различита примерка ентитета истог типа с истим вредностима кандидата за кључ. (На пример за тип ентитета ЛИЧНА_КАРТА, кандидат за кључ је атрибут БРОЈ_ЛИЧНЕ_КАРТЕ). Сложен кључ представља кључ који се састоји из више атрибута, такву врту кључа је потребно креирати само када је то неопходно. Уколико један тип ентитета има више кандидата за кључ, тада бирамо једног од њих и проглашавамо га примарним кључем (На пример примарни кључ за тип ентитета СТУДЕНТ могао би бити атрибут БРОЈ_ИНДЕКСА) [3].Особине Примарног кључа су:

1. једнозначност- два ентитета у скупу никада не смеју имати исту вредност примарног кључа
2. минималност- ако је кључ сложен, он губи своју једнозначност уколико се изостави неки део сложеног кључа.

На слици 12. је приказан графички приказ ентитета и атрибута (примарни кључ и остали атрибути).

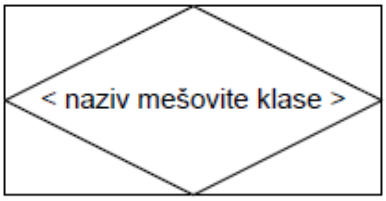


Слика 12. Графички приказ примарног кључа

Веза се може успостављати између два или више типова ентитета (потребно је поседовање одређених односа између ентитета). Графички приказ везе је приказан на слици 13.

Слика 13. Графички приказ везе

Мешовита класа ентитет (Слика 14.) се уводи како би се решио проблем представљања везе између везе, врста везе се представља као класа ентитета и онда се може дефинисати веза између ваза.



Слика 14. Графички приказ мешовите класе

Постоји неколико врста степена веза (кардиналности):

1. Један-према-један (1:1) – код овог типа везе елемент првог скупа може бити повезан са највише једним елементом другог скупа. Истовремено и сваки елемент из другог скупа може бити повезан само са једним елементом из првог скупа.

2. Више-према-један (M:1) – код овог типа везе сваки елемент из првог скупа може бити повезан са највише једним елементом из другог скупа. Истовремено сваки елемент из другог скупа може бити повезан са више елемената првог скупа.

3. Један-према-више (1:N) – један елемент из првог скупа може бити повезан са више елемената из другог скупа и истовремено један елемент из другог скупа може бити повезан са само једним елементом из првог.

4. Више-према-више (M:N)– сваки елемент из првог скупа може бити повезан са више елемената из другог скупа и истовремено сваки елемент из другог скупа може бити повезан са више елемената из првог.

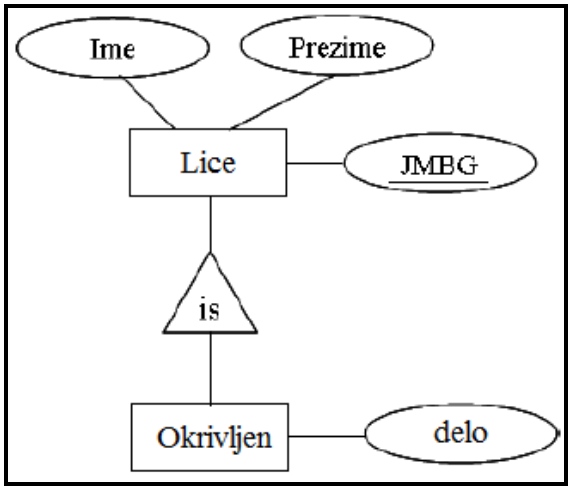
Постоји још врста веза, као што је **унарна веза** која представља везу ентитета са самим собом. Веза може бити и **тотална** или **парцијална**. Дефиниција тоталне везе је да, један тип ентитета има тоталну везу, ако сваки ентитет тога типа учествује у бар једој вези, у супротном веза је парцијална. Постоје и **специјалне везе** којима се дефинише: 1.Егзистенцијална зависност (ова веза је физичка и означава се са E), 2.Идентификациона зависност (ова веза се означава са I, овде важи правило да је кључ једног ентитета део кључа слабог), 3.Идентификациона-егзистенцијална зависност (представља комбинацију претходне две ставке и означава се са E&I).

У оквиру специјалних веза појам слабог ентитета (Слика 15.) представља врсту ентитета који је зависан од другог ентитета, односно таквом ентитету за дефинисање примарног кључа су потребни примарни кључеви других ентитета (спољашњи кључеви), такође могући је настанак слабог ентитета заменом вишеструких веза бинарном. Графички приказ везе између таквих ентитета представљен је на слици Слици 16.

Слика 15. Графички приказ слабог ентитета

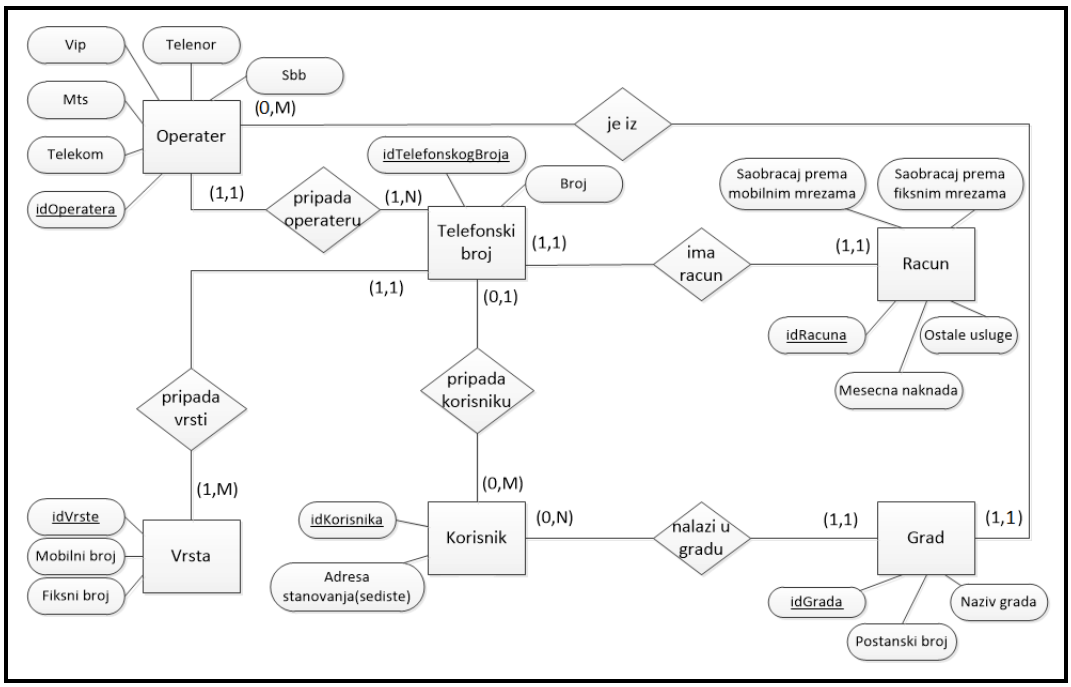
Слици 16. Графички приказ ентитета, слабог ентитета и ознаке везе између њих

Зависност подтипа представља зависност ентитета и ентитета подтипа, ова веза се означава са троуглом са ознаком S у њему (може се наћи и ознака „ is“ или „is a“) (Слика 17). Постоји још један тип везе позната као зависност предхођења**,** таква веза се означава са P, дакле, потребно је да један ентитет предходи другим да би на била могућа.



Слика 17. Графички приказ примера зависности подтипа

На основу детаљне анализе ентитета, атрибута, кључева, веза и кардиналности може се креирати ER дијаграм (као на Слици 18.) који представља основу сваког ER модела базе података. Дијаграм на Слици 18. представља ER дијаграм базе телефонског именика, кога ћу и касније користити у четвртом поглављу за израду модела у програмском пакету Visio.



Слика 18. Интегрисани приказ кардиналности и веза између ентитета (ER дијаграм)

3. Visio – као основни концепт моделирања

Microsoft Visio програмски пакет представља софвер који се користи за креирање графикона, цртежа за организацију простора (просторних планова), мрежних дијаграма али и логичких модела база података и дијаграма. Налази примену и у много другим областима захваљујући шаблонима које је могуће преузети са сајта произвођача. Што се тиче база, у основи Visio подржава три концепта: 1. Извлачење података из постојаће базе како би се створио нови модел (Reverse Engineering), 2. Прављење нове базе која је заснована на одговарајућем моделу и 3. Увожењем (importing) модела.

Reverse Engineering процес, ова некада тешка путања извлачења података из постојеће базе је знатно олакшана захваљујући коришћењу аутоматизованог Reverse Engineering Wizard-a. Који преиспитује систем и алате базе, нуди решења и пријављује проблеме. Овај процес је најлакше користити ако подесимо драјвере базе (Database Drivers), а Visio пакет нам нуди и већи број стандардних драјвера. Све информације везане за овај процес (Reverse Engineering-a) се налазе у Tables and View прозору, док се излази, задаци, евентуалне грешке могу видети у прозору Output.

Увожење подразумева увоз већ моделираног модела базе података из неком другом софтверског пакета у Visio, и то је могуће обавити на врло једноставан начин у одељку Database (главног менија). Visio нуди увожење из софтверског пакета Erwin и Visio Modeler.

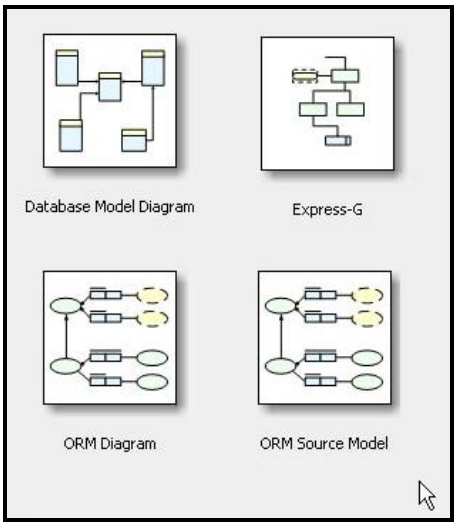
Процес креирања нове базе података (Creating Database) у Visio пакету, корак по корак, биће представљен у поглављу 4.

3.1 Типови модела базе података

Након покретања софтвера добијамо могућност да одаберемо „Visio Datebase template“ („Visio“ шаблон, Слика 19.), професионална верзија овог софтвера нуди следеће модел дијаграме база:

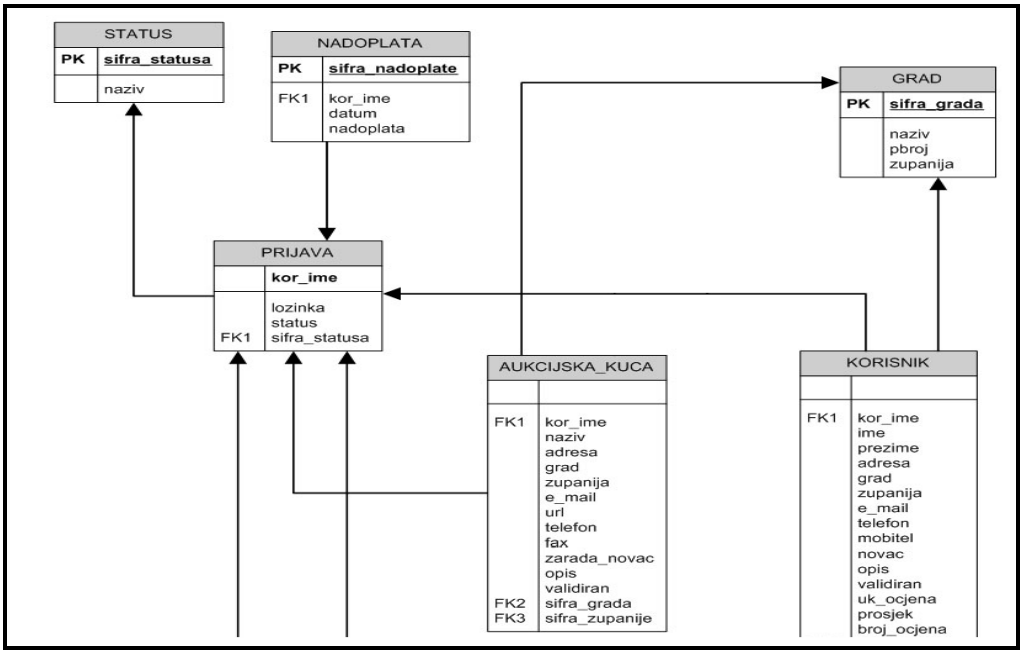
1. ER модел
2. Object Role Model (ORM)
3. Express-G шаблон

Добро је познато да квалитет базе података и апликације у великој мери зависан од дизајна базе података. Савремене базе података су готово незамисливе без основног концептуалног модела базе података и одговарајућег дијаграма, пројектовање као и прављење документације логичког модела базе за релационе и објектно орјентисане базе се врло лако могу савладати.



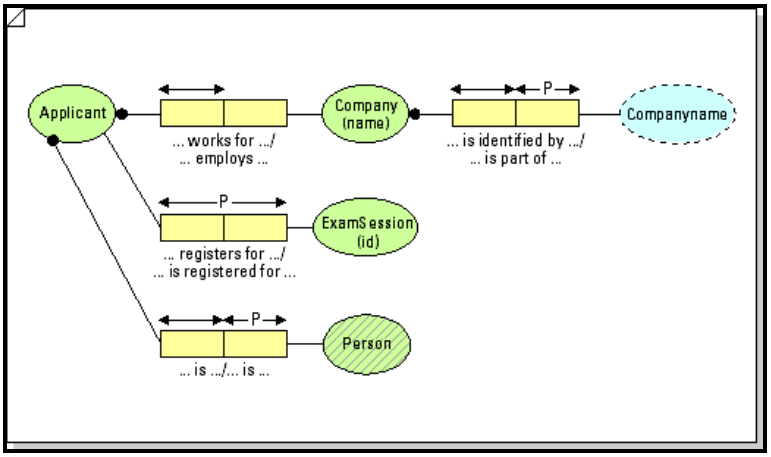
Слика 19. Приказ подржаних дијаграма у „Visio“ пакету

Приликом одабира **ER модела**, „Visio“ омогућава да се креира нова база или да се преправи постојећа. Да би сте извезли модел, потребно је да користите шаблон („Datebase Model template“), који то омогућава поред многих других операције као што је освежавање базе (updating ER sourse model). Овај модел не служи само за цртање дијаграма као остала два, већ за физичко-логичко моделовње кроз табеле, релације, погледе. Без обзира да ли почињете од нуле или преправљате модел, треба узети у обзир да је потребно време да се подесе неке опције за приказ модела у формату који вам је потребан. Дакле, након избора шаблона, вреди посетити опције доступне у прозору „Shapes“. Вероватно најважнија доступна опција је опција везе („Relationship“). Ваш избор ће овде зависити од ваше замисли и одабира модела [5]. На слици 20. је приказан пример ER дијаграма, који представља продукт овог модела.



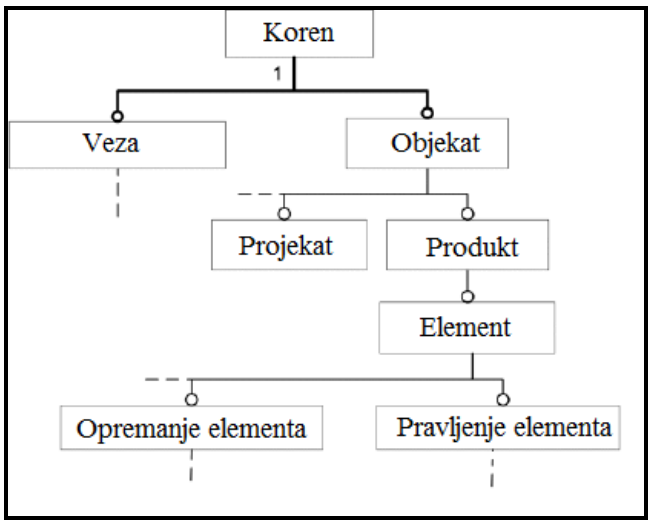
Слика 20. Приказ ER модел дијаграма у „Visio“ пакету

Object role model (Orm) дијаграм чији је пример приказан на Слици 21., захваљујући подршци свих потребних облика (shapes) у пакету који су карактеристични за Object role модел може се врло једноставно креирати дијаграм. Object role model представља приступ семантичком моделирању у погледу објекта, њихових улога, ограничења и релација.



Слика 21. Интегрисани приказ Object role model дијаграма

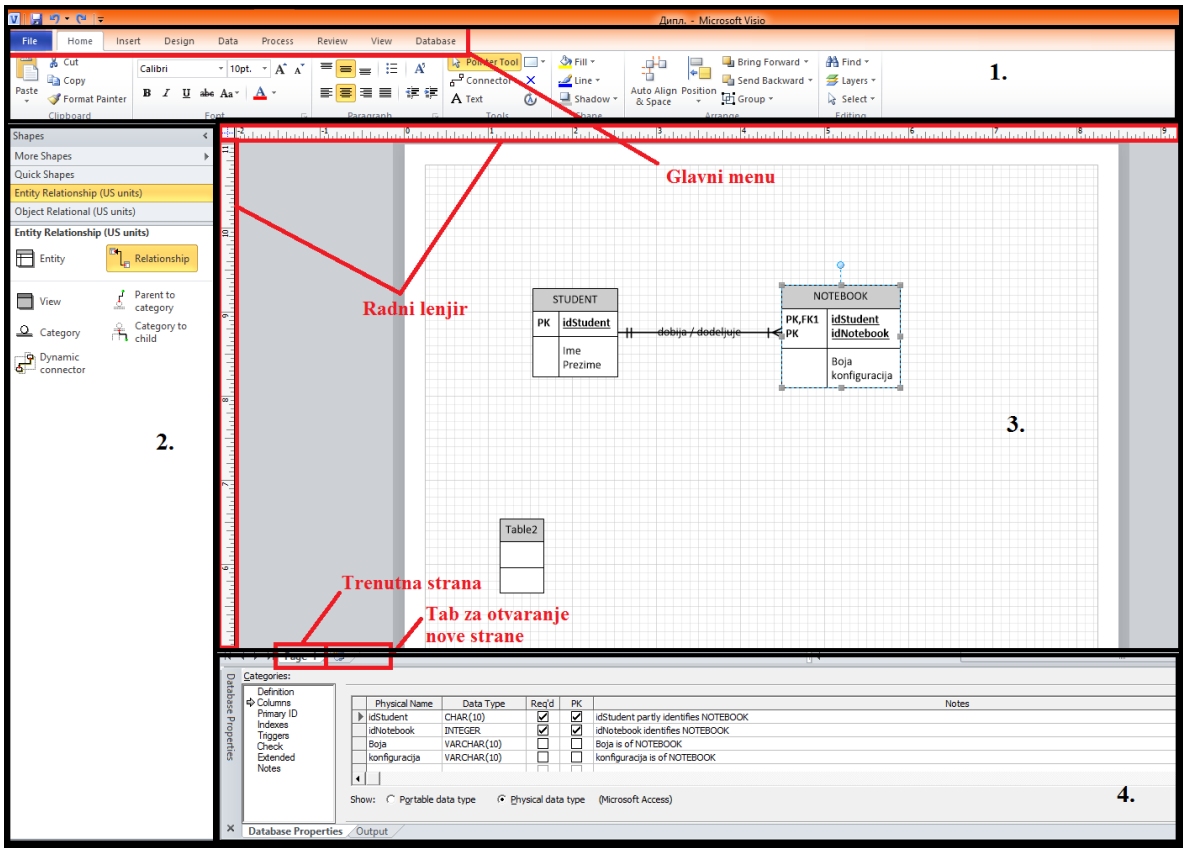
Express-G у „Visio“пакету омогућава креирање дијаграма модела базе помоћу облика и форми у нотацији Express-G модела. Express-G (пример дијаграма на слици 22.) модел се користи при приказивању великих модела информација преко сопсвених релација, веза и структура.



Слика 22. Приказ форме Express-G дијаграма

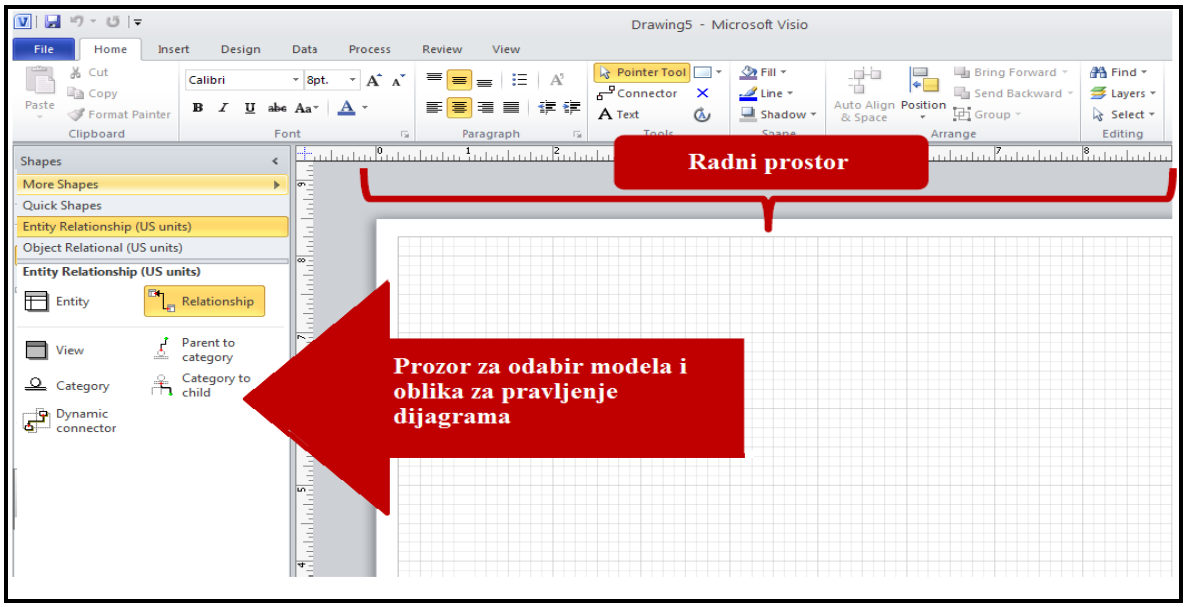
3.2 Упознавање окружења „Visio“ софтверског пакета

У „Visio“ прозору први, горњи део представља мени (Слика 23.), у ком се налазе одељци „File“, „Home“, „Insert“, „Design“, „Data“, „Process“, „Review“, „View“ и одељак „Database“.



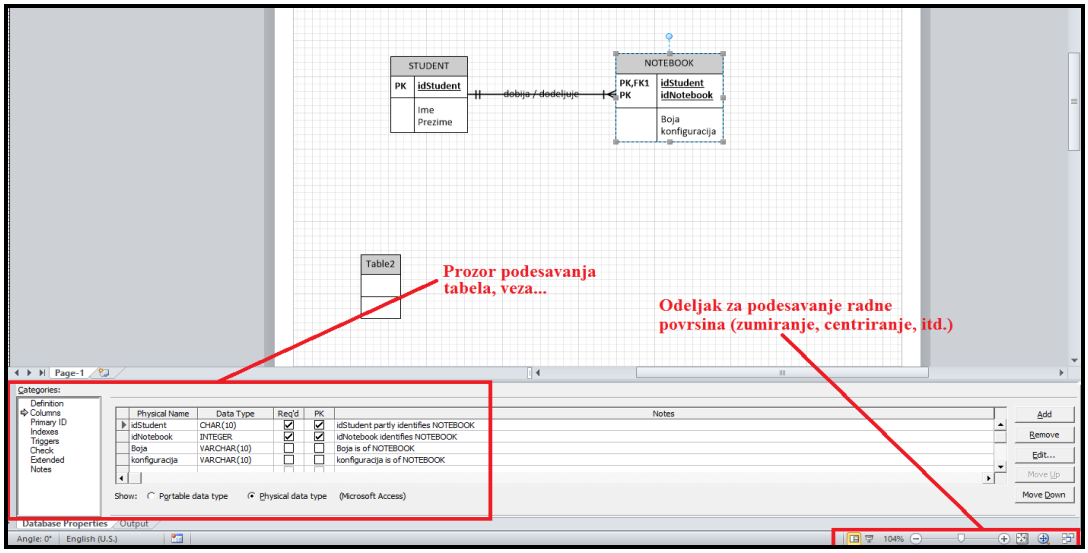
Слика 23. Приказ радног окружења у „Visio“ пакету

Одељак „File“ служи за подешавање основних функција нашег отвореног фајла, као што је преглед функције за штампање, слање, пружање информација о документу, о сњимању и много других опција. Одељак „Home“ поседује функције за тражење текста, његово обраду, али и алат као што је „Connector“ између табела. Одељак „Insert“ се користи за убацивање нових страна, убацивања слика, клипова, симбола, хиперлинкова, текстова и сличног. Одељак „Design“ нуди опције за мењање позадине, попут додавања боја, мењања димензија радног папира (простора), оквира , додавања ефекта и др. Овај одељак садржи широк спектар дизајна за моделирање дијаграма. Одељак „Data“ се користи за рад са подацима, убацивање или повезивање података из других софтвера, приказивање података, убацивање аутоматизоване легенде и др. Одељак „Process“ пружа могућности провере валидности дијаграма, SharePoint увожења и извожења, додавања хиперлинкова облицима и сличног. Одељак „Review“ поседује опције за обележавање, рад са коментарима, језицима, језичком исправношћу и др. Одељак „View“ користи се за одабир начина приказивања радног простора (приказивање на целом екрану, увеличавање, отварања новог прозора и др.), рад са макро наредбама и осталог. Одељак „Database“ нуди опције за рад са моделом (базом), управљање драјверима базе, могућност отварања/затварања видљивости веза, делова табела и општих подешавања везаних за базу.



Слика 24. Приказ радног окружења у „Visio“ пакету

Други, леви део прозора омогућава брз и једноставан приступ различитим категоријама облика за моделирање базе података, само једноставним превлачењем одговарајућег облика на радни простор (који представља трећи, централни део, Слика 23 и Слика 24.) добијамо рецимо табелу којој можемо потом доделити атрибуте и остале њој својствене елементе.



Слика 25. Приказ радног окружења у „Visio“ пакету

Четврти, доњи део прозора садржи основне информације као и део за подешавања базе, табеле, везе или неког облика у зависности од тога на чему се тренутно ради (шта је селектовано). Овај део такође садржи и пречице за подешавање прозора радне површине и њених елемената попут зумирања (што се може видети на слици 25.), прегледа радне површине на целом екрану и сличног.

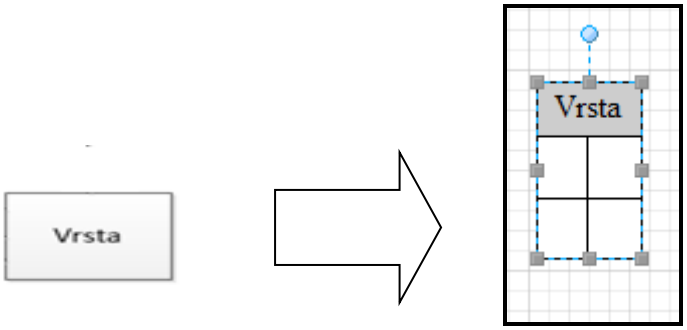
3.3 Шаблони, панели и форме (Shapes)

Visio у себи садржи огроман број прототипова (шаблона) – модела за одређени тип цртежа (Tepmlate). Прототип дефинише одређене карактеристике као што су: низ одговарајућих форми или облика, величина и оријентација страница цртежа, размера, одговарајућих стилова за текст и линије, панела као и начина попуњавања затворених површина. Могуће је мењати ове елементе, али сврха коришћења одређеног прототипа је козистентност цртежа. Цртеж може имати више страница које је могуће гледати као целину. Панели (Stencils) су груписане форме (Shapes), које представљају највећу предност овог софтвера (велика библиотека облика). Приликом прављења новог модела Visio нуди две палете Entity Relationship и Object Relational. Приликом одабира панела треба водити рачуна, рецимо, Entity Relationship поседује „Entity“ форму (користи се за представљање табеле), Relationship конектор (користи се за повезивање табеле које је зависна са табелом од које зависи), „View“, „Category“, „Parent to Category“ и „Category to Child“ форму (које ћу детаљније објаснити у наредним поглављима), уз све ове форме при одабиру друге палете Object Relatonal се добијају „Table Inheritance“ и „Type Inceheritance“ конектори за аутоматско наслеђивање атрибута „родитељ“ од „дете“ табеле или типа.

3.4 Начин представљања ER модел дијаграма у Visio пакету

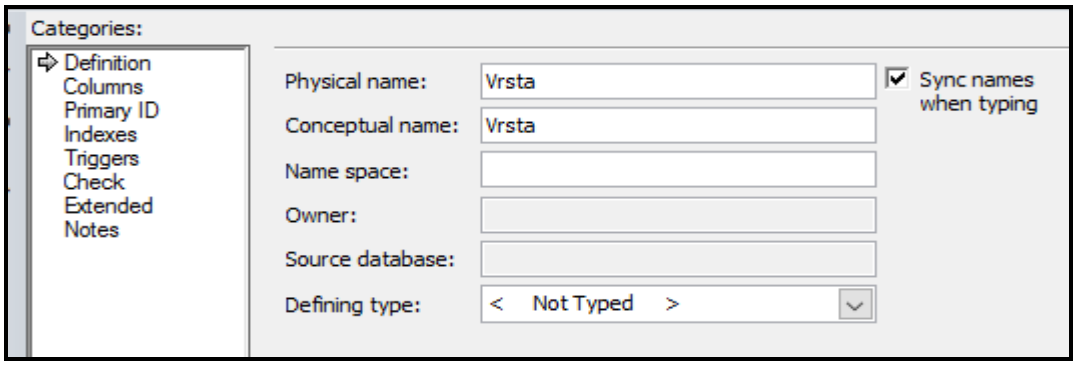
У овом поглављу ћу говорити о начину представљања ER дијаграма, односно ентитета, атрибута, кључева и веза у Visio пакету, док ће детаљни поступак моделирања базе кроз кораке бити приказан кроз пример у следећем поглављу.

Ентитет, као пример узећу ентитет „Vrsta“, његово представљање (Слика 26.) је кроз вид табеле у нашем софтверу, он се добија једноставним превлачењем из палете Entity Relational на наш цртеж.



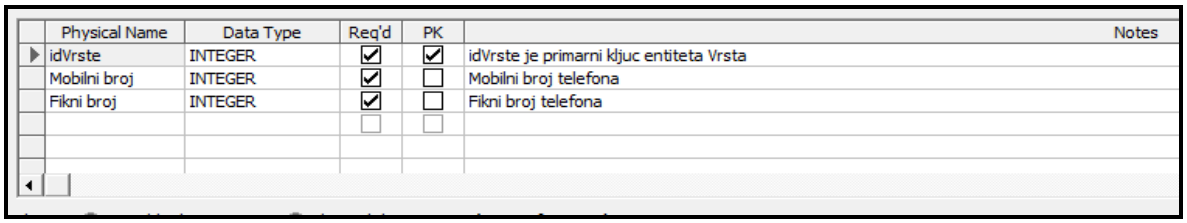
Слика 26. Интегрисани приказ ентитета у „Visio“ пакету

Назив ентитета се додељује (након клика на ентитет) у прозору који се појављује испод, под називом „Categories“ (Слика 27.) и његовом одељку „Definition“, „Physical name“.



Слика 27. Интегрисани приказ додељивања назива ентитету у „Visio“ пакету

Као што се може видети на слици од осталих категорија су ту „Columns“, „Primary ID“, „Indexes“, „Triggers“, „Check“, „Extended“ и „Notes“. Категорија „Columns“ (Слика 28.) се користи за додавање атрибута, примарних кључева, описа атрибута као и типова.



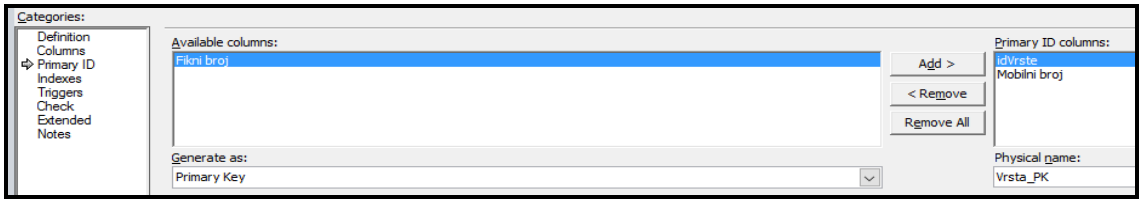
Physical Name	Data Type	Req'd	PK	Notes
idVrste	INTEGER	☑	☑	idVrste je primarni kljuc entiteta Vrsta
Mobilni broj	INTEGER	☑	☐	Mobilni broj telefona
Fikni broj	INTEGER	☑	☐	Fikni broj telefona

Слика 28. Интегрисани приказ примарног кључа и осталих атрибута ентитета Vrsta у „Visio“ пакету

Атрибути, додавање атрибута је врло једноставно, у колони „Physical Name“ се дају имена атрибутима, потом у колони „Date Type“ типови података атрибута у зависности од њихиве сврхе. Колона „Req'd“ представља скраћеницу речи required, то се односи на обавезу атрибута (да ли су неопходни или би ентитети функционисали и без њих).

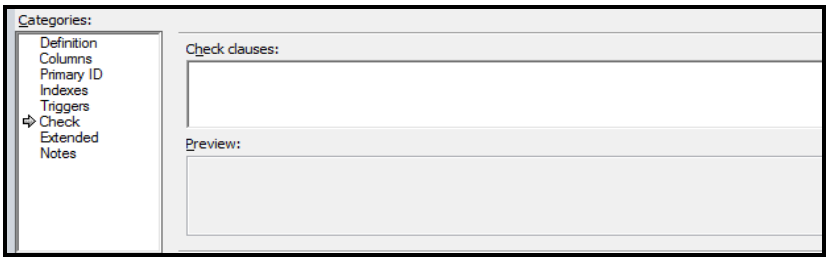
Примарни кључ, колона „PK“ (Primary Kay) служи за одређивање примарног кључа, дакле једноставним чекирањем квадрата неког атрибута, он постаје примарни кључ. Један ентитет може имати више примарних кључева.

Категорија „Primary ID“ (Слика 29.) користи се за лакше управљање атрибутима, рецимо да желимо одређене атрибуте да обришемо или поставимо као примарне кључеве, то би могли урадити овде једноставним кликом.



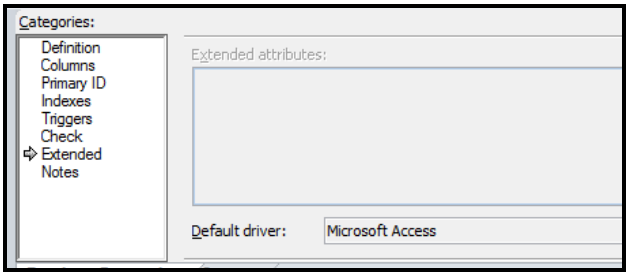
Слика 29. Интегрисани приказ категорије „Primary ID“ у „Visio“ пакету

Категорија „Indexes“служи за индексирање, индексирање побољшава брзину и перформансе базе при покретању упита. Индексирање ће заправо помоћи систему базе података да нађе и сортира записе брже, софтвер аутоматски додељује индексе записима, али у „Visio“програмском пакету постоји могућност да се ово и самостално одради. Категорија „Triggers“ се корити код SQL кода као окидач за покретање при одређеним догађајима у бази. Приликом клика на категорија „Check“ (Слика 30.) добијамо два поља, прво, „Check clauses“ у коме се може видети листа свих постојећих ограничења (њихових имена), уколико постоје, док се у другом пољу „Preview“ види кратак опис ограничења. „Check“ категорија се дакле користи за проверу клаузула, постављања ограничења на одређену табелу или колону атрибута.



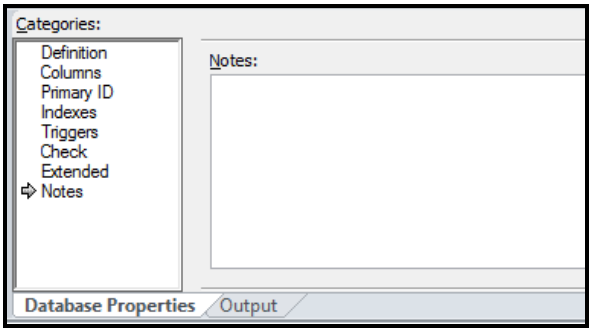
Слика 30. Интегрисани приказ категорије „Check“ у „Visio“ пакету

Категорија „Extended“ (Слика 31.), се користи ако ваша база поседује ткз. „extended“ атрибуте, овде се они требају подешавати. Такође, овде се може видети одабрани „Default driver“, који се може променити у одељку „Detabase->Database Drivers“ који се налази у главном табу прозора.



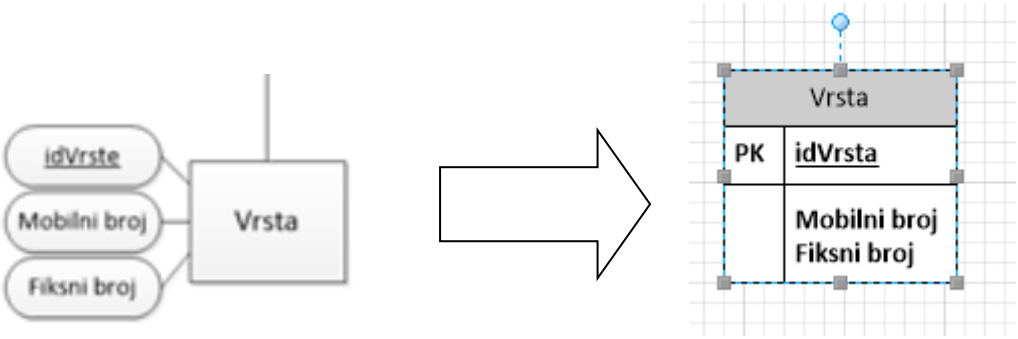
Слика 31. Интегрисани приказ категорије „Extended“ у „Visio“ пакету

„Notes“(Слика 32.) категорија служи за описивање ентитета, садржаја ентитета, коментара и осталог.



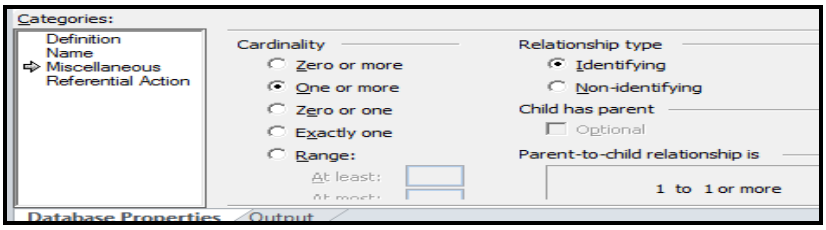
Слика 32. Интегрисани приказ категорије „Notes“ у „Visio“ пакету

Када се завршило креирање ентитета са свим атрибутима, одређивањем примарног кључа и проласком кроз све потребне категорије, добија се ентитет приказан као на слици 33.



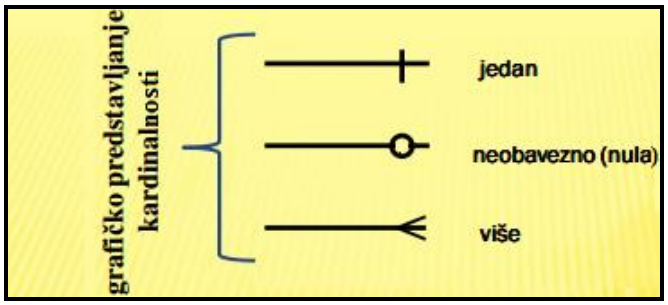
Слика 33. Интегрисани приказ ентитета (као табела) у „Visio“ пакету

Везе, за стварање веза користи се Relationship конектор, када се креира веза између две табеле (ентитета) у „Visio“ пакету, дете-табели се додају атрибути (примарни кључеви) родитеља-табеле. Након превлачења и постављања Relationship конектора између табела добијамо могућност одређивања кардиналности, правила референцијалног интегритета, које табеле наслеђује које особине као и примарне кључеве. На Слици 34. је приказан прозор подешавања везе између табела.



Слика 34. Приказ прозора са подешавањима веза у „Visio“ пакету

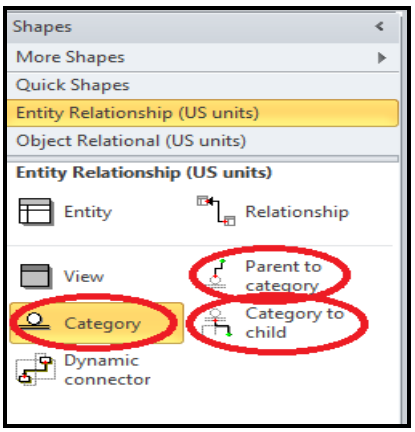
Кардиналност, графички приказ представљања кардиналности је приказан на слици 35. У наредном поглављу ћу објаснити коришћење степена релације кроз примере.



Слика 35. Графички приказ кардиналности у Visio пакету

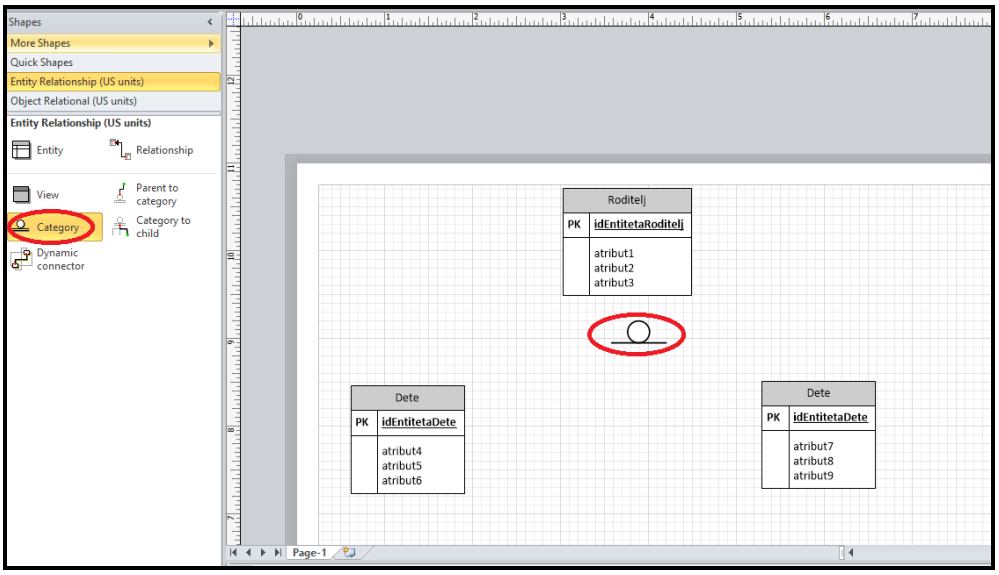
3.5 Категоризација табела и дефинисање погледа базе (View)

„ Entity Relationship“ форма поседује сем могућности превлачења ентитета и веза, могућност категоризације табела и дефинисање погледа. Категоризација табела олакшава прављење већег броја табела које деле заједничке примарне кључеве и атрибуте, „Visio“ нуди олакшавајућу опцију тако да превлачењем облика за категоризацију правимо јединствену категорију, која сем примарног кључа и атрибута поседује и дискриминатор колону која омогућава „Visio“ софтверу да препозна којој категорији припада табела. „Category shape“ су категорисане табеле у „Visio“ моделу, њихово конектовање са родитељ-табелама се врши превлачењем „Parent to Category“ иконе (Слика 36) из прозора „Shapes“ на радни простор (у наш направљен модел), док се дете-табела повезује превлачењем иконе „Category o child“ (Слика 36).

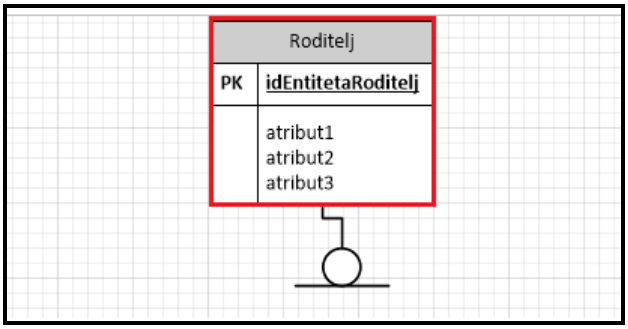


Слика 36. Приказ „Shapes“ прозора у „Visio“ пакету

Прво што је потребно урадити да би направили категорију, након креирања ентитета (табела) са атрибутима, је превлачење „Category“ иконе у радан простор-наш модел (Слика 37). Након овог корака треба се превући „Parent to category“ конектор, такође из прозора „Shapes“ и један његов крај везати за родитељ-табелу, а други за графички симбол категорије. Треба водити рачуна да се приликом овог повезивања (након узимања једног краја конектора левим кликом миша) нађе одговарајући положај, док табела не поцрвени (Слика 38.), тек потом пустити леви клик миша на табелу (исти поступак важи и за повезивање са категоријом).

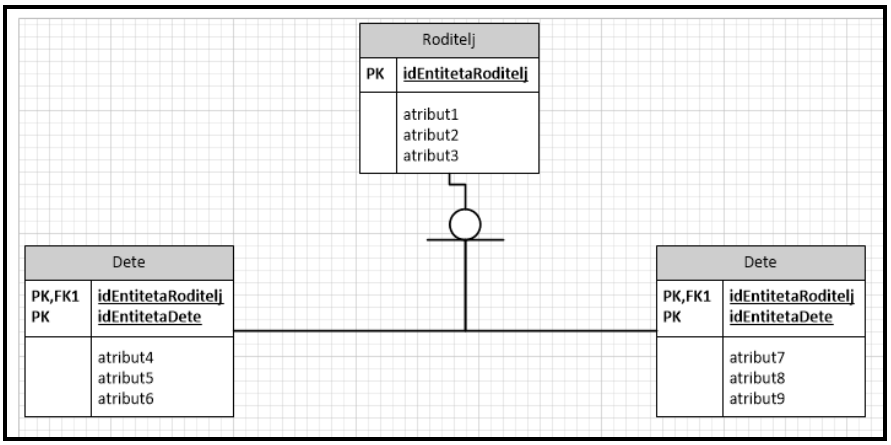


Слика 37. Приказ креирања категорије у „Visio“ пакету



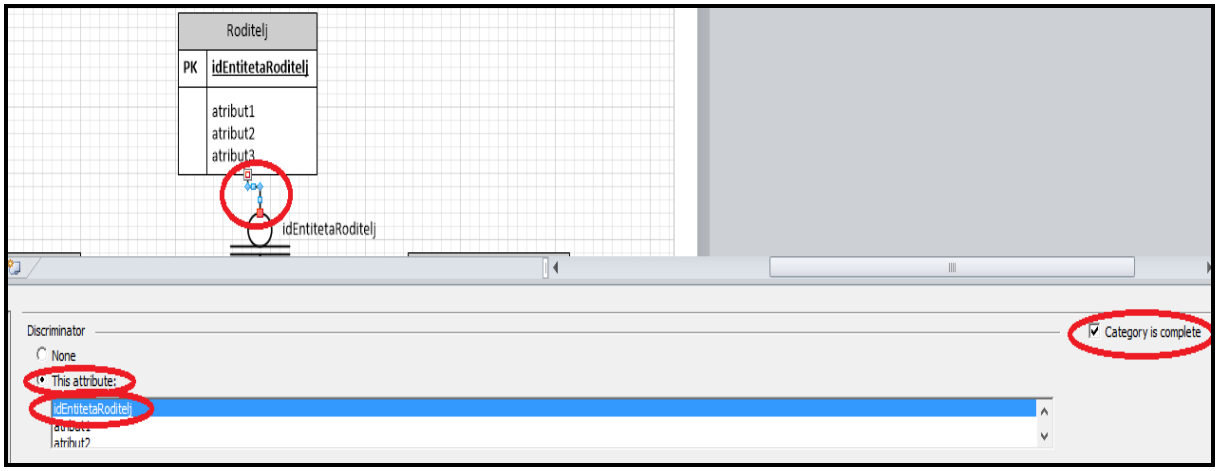
Слика 38. Приказ повезивања категорије и родитељ-табеле у „Visio“ пакету

Следећи корак је повезивање свих дете-табела са категоријом (симболом) у моделу преко конектора „Category to child“(Слика 39). Примећује се да софтвер у дете-табелама, аутоматски ствара примарне и спољне кључеве, због табеларних међу односа.



Слика 39. Приказ повезивања категорије и дете-табеле у „Visio“ пакету

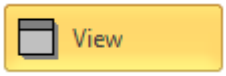
Потребно је још одредити дискриминатор, то се ради кликом на везу између категорије и родитељ-табеле (у прозору за њихова подешавања „Database Properties“), опцијом „This Attribute“, селекцијом атрибута и чекирањем опције „Category is complete“, што је приказано на слици 40.



Слика 40. Приказ подешавања при креирању категорије у „Visio“ пакету

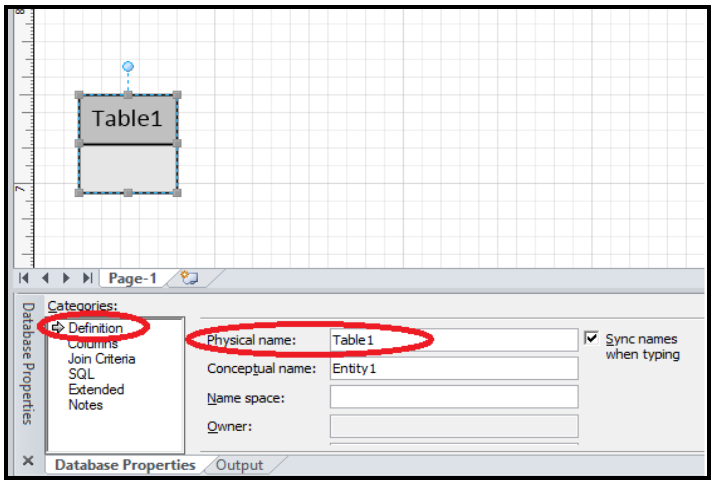
Дефинисање погледа базе података (View)

Као што сам већ напоменуо, Entity relationship форма садржи и команду за дефинисање погледа (View) базе података, ово је корисно због лакшег посматрања одабраних података, а притом да не реметимо сам концепт креираног модела. Приликом коришћења ове опције, софтвер генерише SQL код који га описује. Први корак је превлачење иконе „View“ (Слика 41), који се налази у прозору „Shapes“, на модел.

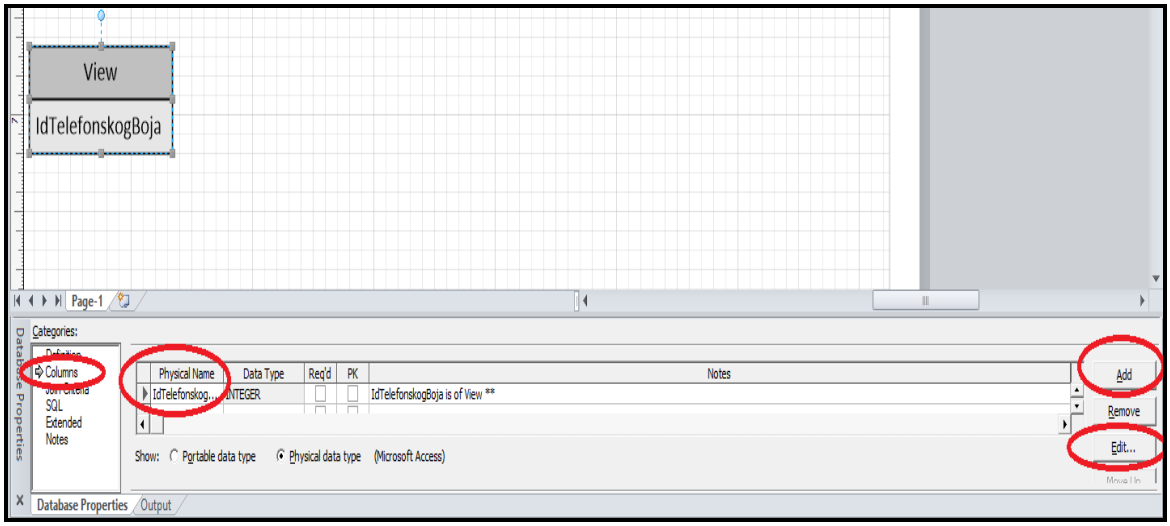


Слика 41. Приказ иконе „View“ у „Visio“ пакету

Након клика на табелу „View“, у прозору „Database Properties“, категорији „Definition“, у пољу „Physical name“ (Слика 42.) дајемо име, потом у категорији „Columns“(Слика 43), делу „Physical Name“ дајемо назив новим колонама (поред „Physical Name“ колоне, овде постоји и „DataType“ , која служи за задавање типа података, као и колона за означавање примарног кључа и бележака), кликом на „Add“, софтвер задаје стандардно име колони (на основу подешавања у прозору „Database Modeling Preferences“). Следећи корак је навођење извора за ту колону, кликнемо на „Edit“.

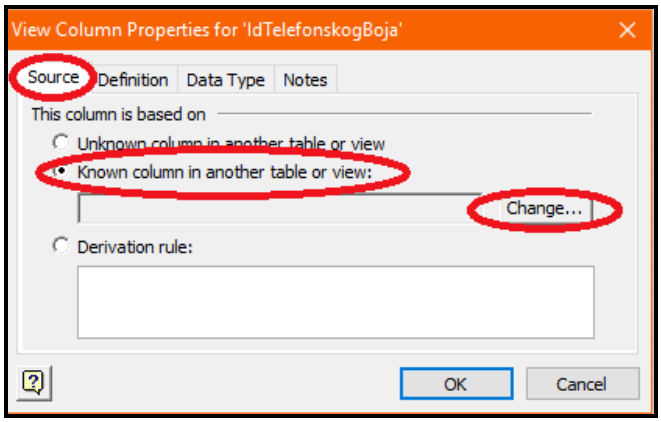


Слика 42. Приказ прозора„Database Properties“ у „Visio“ пакету

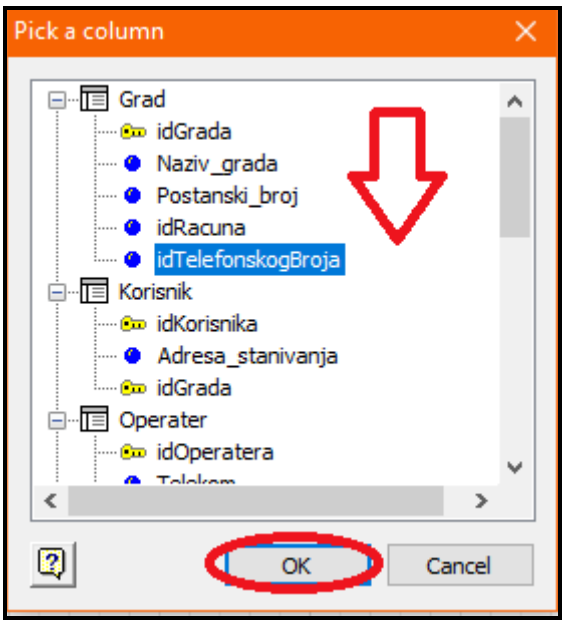


Слика 43. Приказ прозора„Database Properties“ (дела „Columns“) у „Visio“ пакету

Потом, у прозору „View Column Properties“, на „Sourse“ одељку, потребно је кликнути на „Known Column in Another Table or View“ опцију (Слика 44), па Change. У ново отвореном прозору „Pick a Column“(Слика 45.) требамо одабрати колону са листе табела и погледа и кликнути на „Ок“. Брисање колоне је једноставно у категорији „Columns“, одабиром опције „Remove“, одабрана колона се брише. Табела „View“ се брише идентично као и све остале табеле, кликом на одабрану табелу и притиском тастера „Delete“на тастатури. Одабиром категорије „SQL“, прозора „Database Properties“, се може видети SQL код ког је софвер претходним корацима аутоматски генерисао.



Слика 44. Приказ прозора „View Column Properties“ у „Visio“ пакету



Слика 45. Приказ прозора „Pick a column“ у „Visio“ пакету

Категорија „Join Criteria“, у прозору „Database Properties“(који се односи на табелу погледа), пружа могућност додавања критеријума, једноставним одабиром колона које имају неке заједничке податке и кликом на „Add“. Део „Criteria“ ове категорије приказује све креиране критеријуме.

3.6 Мењање кода

Прозор „Code“ (чекирањем опције „Code“ у менију „Database”) се користи за креирање, мењање или брисање старих кодова. У овом прозору сваки унос поседује придружен назив табеле или погледа која га поседује. „Code“ прозор приказује све кодове који су доведени у везу са моделом наше базе укључујући и кодове који су дошли са базом (код увежених база). Приликом коришћења „Code“опције препоручљиво је и коришћење mirror фајла [4], који се користи за чување кода. Овај фајл се налази ван Visio програмског пакета, на локацији коју ми одаберемо. Mirror

фајл дефинишемо тако што у прозору „Code“, ког смо отворили на горе поменут начин, кликнемо на „New“, потом у новоотвореном прозору одаберемо одељак „Properties“, где је прво потребно дати име Mirror фајлу у пољу „Name“, након тога, испод овог поља се налази опција „Browse“ која се користи за навођење локације нашег фајла. У прозору „Code*“* се налазе две категорије „Global Code“ и „Local Code“. Категорија „Global Code“садржи кодове и све функције, процедуре за чији рад је потребна функција система (функционална зависност од система). Друга кетегорија, „Local Code“, садржи све кодове који су везани за табелу, погледе или колоне (check clauses и triggers). Visio програмски пакет подржава следеће типове кодирања база података:

• ***Stored procedures*** **–** За прављење процедуре потребно је у прозору „Code“ одабрати категорију „Global Code“ потом опцију „New“. У новоотвореном прозору „Code Editor*“* можете користећи SQL направити процедуру (у одељку Body) а можете и на др. Начин, одаберите одељак „Properties“, затим у пољу „Name“ доделите име и на крају одредите један тип од понуђених: Stored Proc, Function или Raw DDL.

• ***Check clauses*** – Овај тип кодирања се приказује само у „Local Code“категорији (прозор „Code*“*), ове услове провере је могуће креирати у прозору подешавања „Database Properties“, категорији „Check“, након клика на опцију „Add“. Након чега се отвара прозор „Code Editor“, где се у делу „Body“ креира SQL код, а у одељку „Properties“ даје жељени назив.

• ***Triggers*** **–** Овај тип кодирања се такође приказује само у „Local Code“категорији (прозор „Code*“*). Окидаче је могуће направити одабиром одговарајуће табеле (једноструким левим кликом на одг. табелу), потом се у прозору „Database Properties“, бира категорија „Triggers“ и опција „Add“. Након чега се отвара прозор „Code Editor“, где се у делу „Body“ креира SQL код, а у одељку „Properties“ даје жељени назив.

• ***View SQL Code*** **–** Преглед као и мењање SQL кода код погледа (View) табеле, се врши једноставним одабиром одговарајуће „View“ табеле, потом је у прозору „Database Properties“ потребно одабрати категорију „SQL“ и деселектовати опцију под називом „Auto-Generated“.

Дијалог „Code Editor“ такође нуди и омогућава неколико корисних алата и пречица, да би смо омогућили овај процес мењања кодова потребно је да кликнемо на икону „Windows Properties“ која се налази у алатима Code editor-a која отвара „Windows Properties dialog box“.

3.7 Подешавање моделирања

Како би изменили подешавања процеса самог моделирања потребно је одабрати у менију одељак „Database“ и кликнути на опцију „Modeling Preferences“. Новоотворени прозор садржи два одељка „Logical Diagram“ и „Logical Misc“. Одељак „Logical Diagram“ садржи следеће опције за подешавања:

• ***When Removing an Object from the Diagram*** – Овде се нуде опције за одабир поступка са објектом, рецимо да ли ће објекат аутоматски бити избачен из модела или ће бити питам корисник.

• ***Show Relationship After Adding Table to Diagram*** – Чекирањем ове опције врши се приказивање релација између управо додатих типова.

• ***Show Relationship After Adding Type to Diagram*** – Чекирањем ове опције врши се приказивање релација између управо додатих табела.

• ***Sync Conceptual and Physical Name in New Tables and Columns When Typing*** – Чекирањем ове опције врши се синхронизација промена у заједничким именским пољима

Одељак „Logical Misc“ садржи следеће опције за подешавања:

• ***FK Propagation*** – Чекирањем опције „Propagate on Add“софтверу дајемо право, чим се повеже Relationship конектор, да аутоматски прави везу између родитељ-дете табеле преко спољашњег кључа (foreign key). Чекирањем опције „Propagate on Delete“ врши се уклањање везе која је направљена преко спољашњег кључа, уколико се Relationship конектор уклони.

• ***Name Conflict Resolution*** – Овде имамо могућност да одаберемо шта ће се догодити уколико се у моделу појави страни кључ који поседује исти назив као и колона у дете-табели.

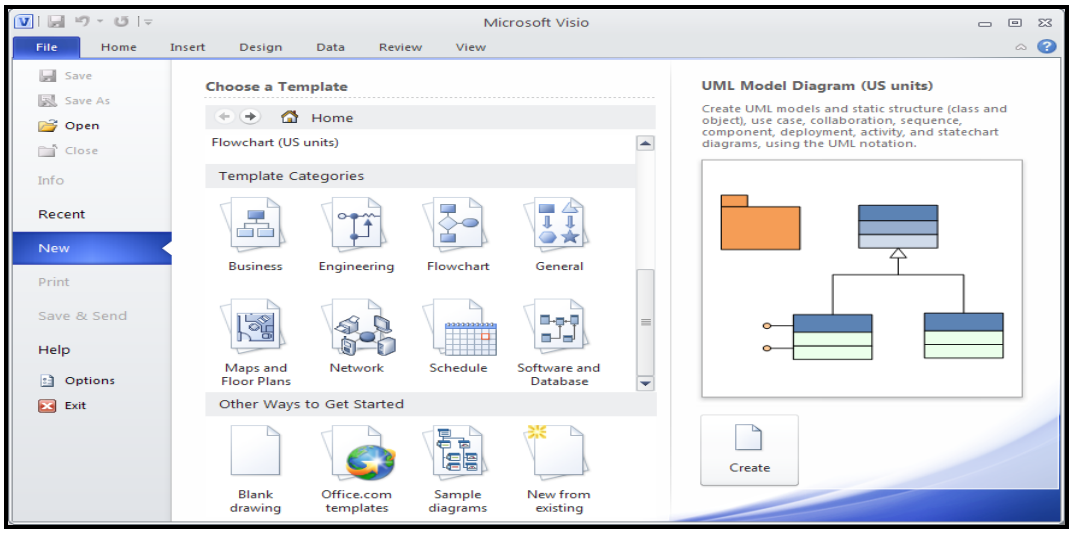
• ***Default Name Prefixes***–Користи за одабир префикса при коришћењу концептуалног имена у моделу.

• ***Default Name Suffixes*** – Користи за одабир суфикса који додајемо default префиксу концептуалног имена у моделу.

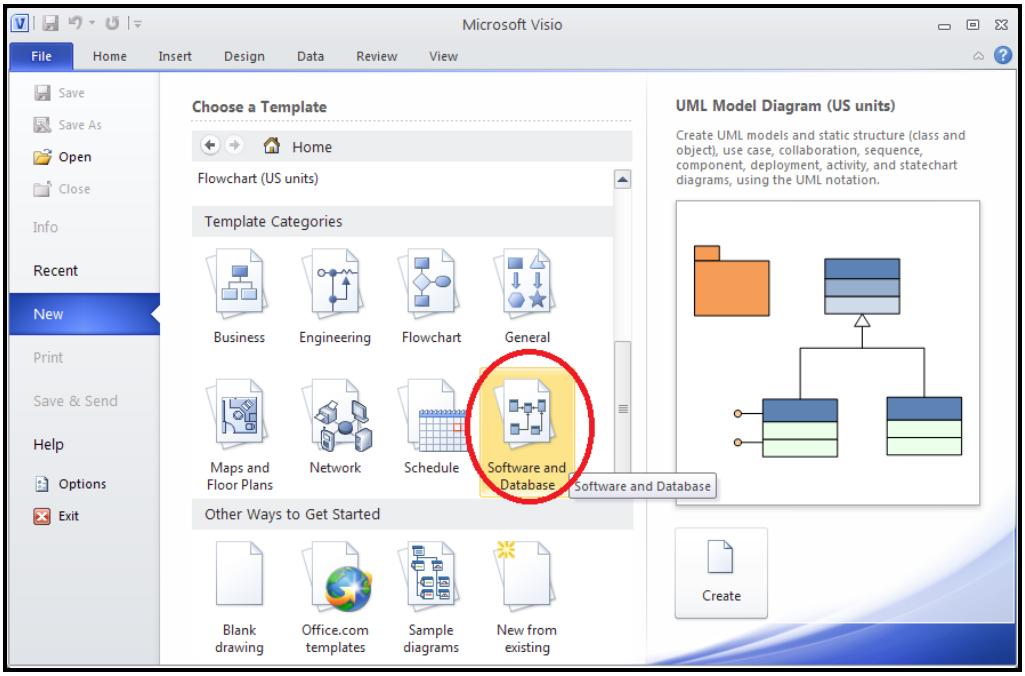
• ***FK Name Generation Option*** – Наводимо објекте који ће се користити при аутоматском додељивању имена спољашњих кључева.

4. Пример коришћења Visio пакета

У овом поглављу ћу објаснити кораке за креирање новог модела базе података у нашем Visio софтверу. Прво што је потребно урадити након покретања Visio пакета (кликом на: Start->All Programs->Microsoft Office->Microsoft Visio 2010) у добијеном прозору (Слика 46.) је одабир шаблона, кликом на врсте шаблона „Software and Database“ (Слика 47),

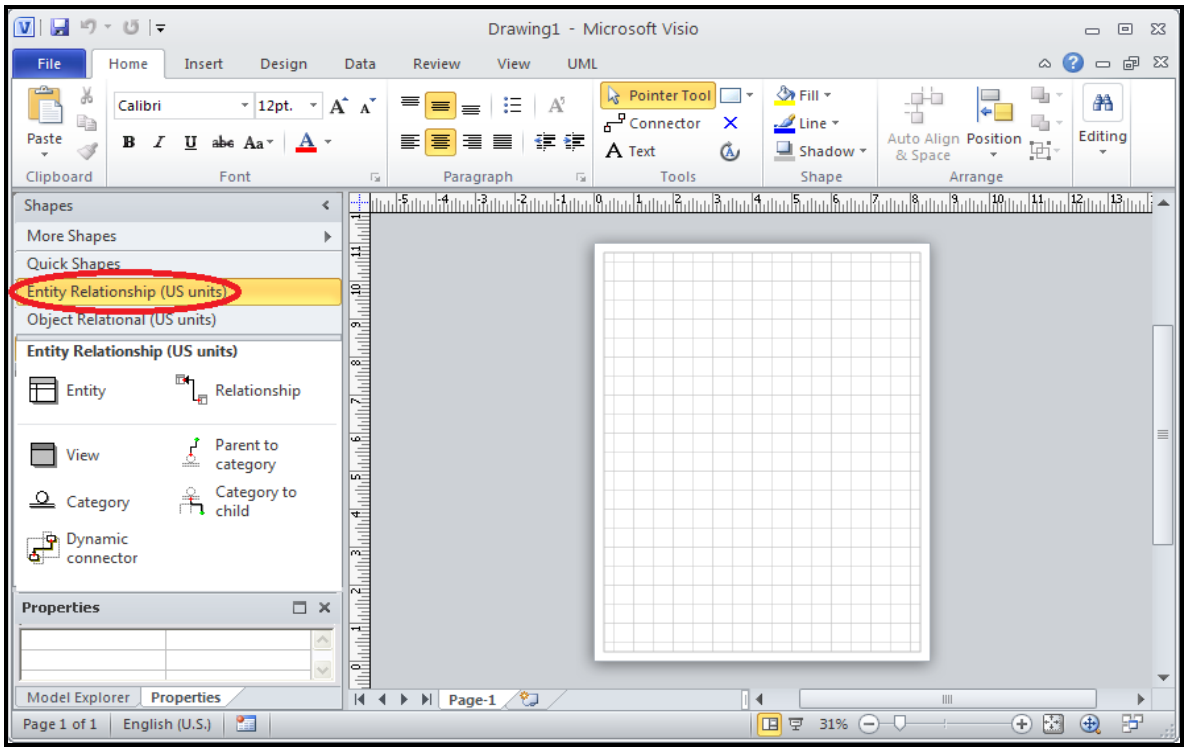


Слика 46. Приказ прозора за одабир шаблона у „Visio“ пакету



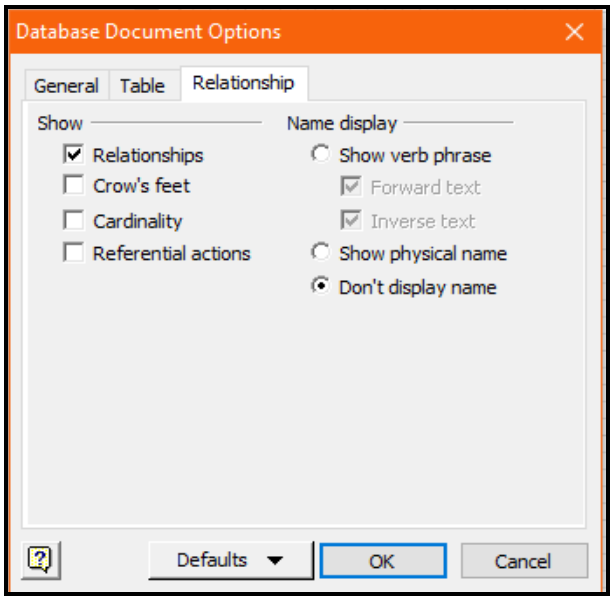
Слика 47. Приказ одабира шаблона „Software and Database“ у „Visio“ пакету

Потом кликом на „Database Model Diagram“ добијамо радно окружење за моделирање базе података (Слика 48), овај прозор (његови делови) су детаљно објашњени у претходном поглављу. Након тога, потребно је одабрати у левом делу прозора, у одељку „Shapes“, „Entity Relationship“.



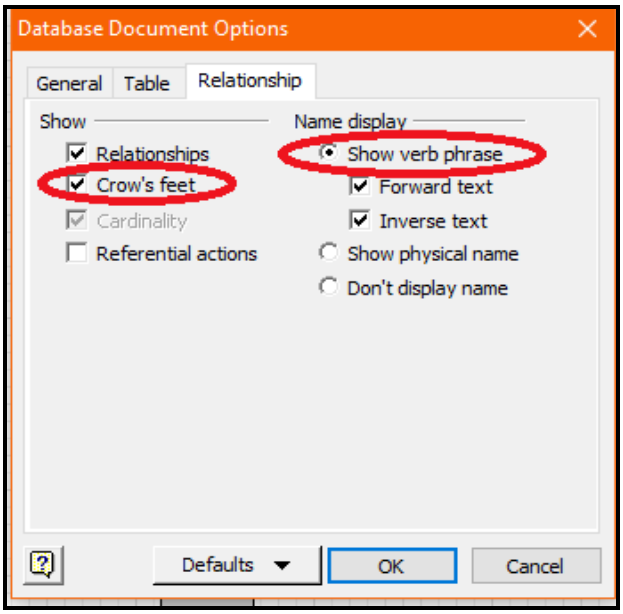
Слика 48. Приказ радног окружења у „Visio“ пакету

Сада је потребно променити начин приказивања веза, кликом на одељак „Database“ главног менија, затим на „Display Option->Relationship“ добија се прозор као што је приказан на слици 49.



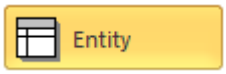
Слика 49. Приказ прозора „Detabase Document Options“ у „Visio“ пакету

Потребно је променити стандардан начин приказивања веза у пакету, омогућити опције „Crow’s feet“ и „Show verb phrase“ (Слика 50.), Након овог поступка можемо изаћи из овог прозора.

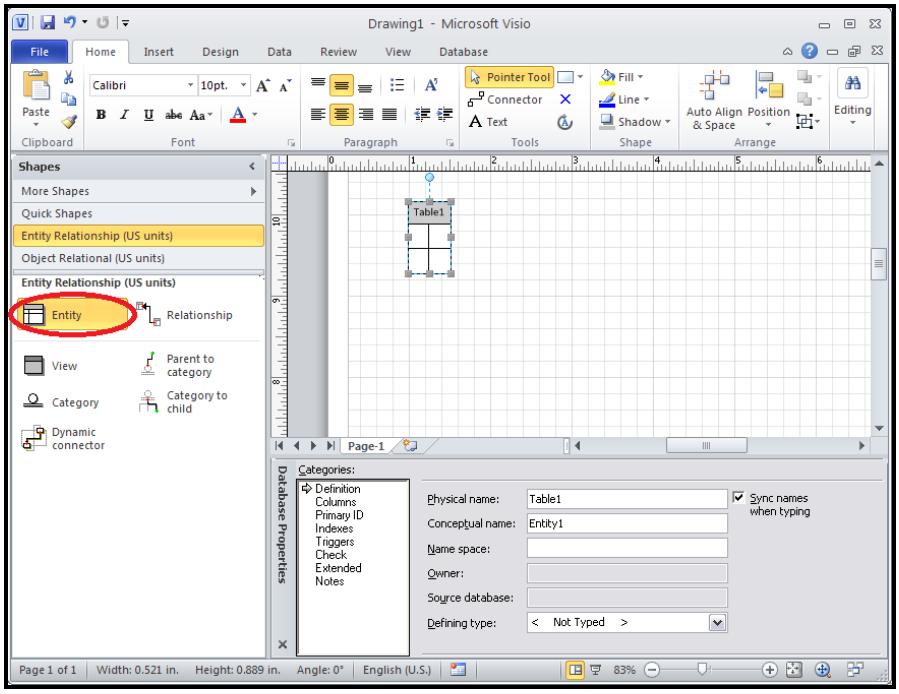


Слика 50. Приказ прозора „Detabase Document Options“ у „Visio“ пакету

Као што сам већ напоменуо ентитет се у нашем софтверу приказује кроз табелу, он се може убацити у наш модел базе једноставним превлачењем иконице (Слика 51 и 52.) у радни простор (добијена табела је приказана на слици 52)

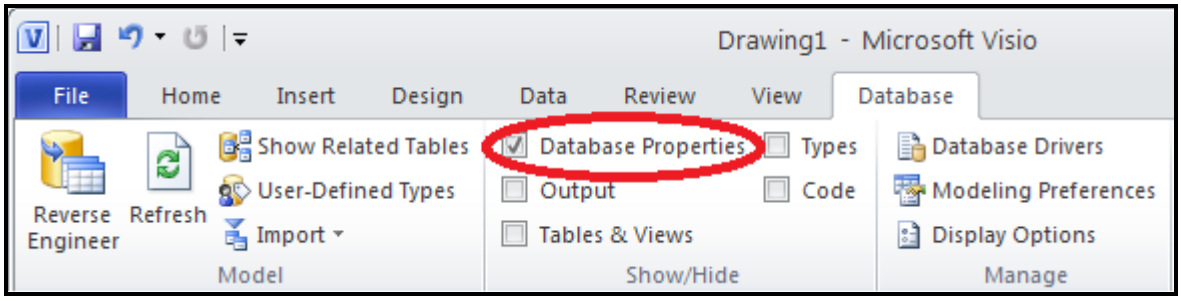


Слика 51. Приказ иконице Entity у „Visio“ пакету



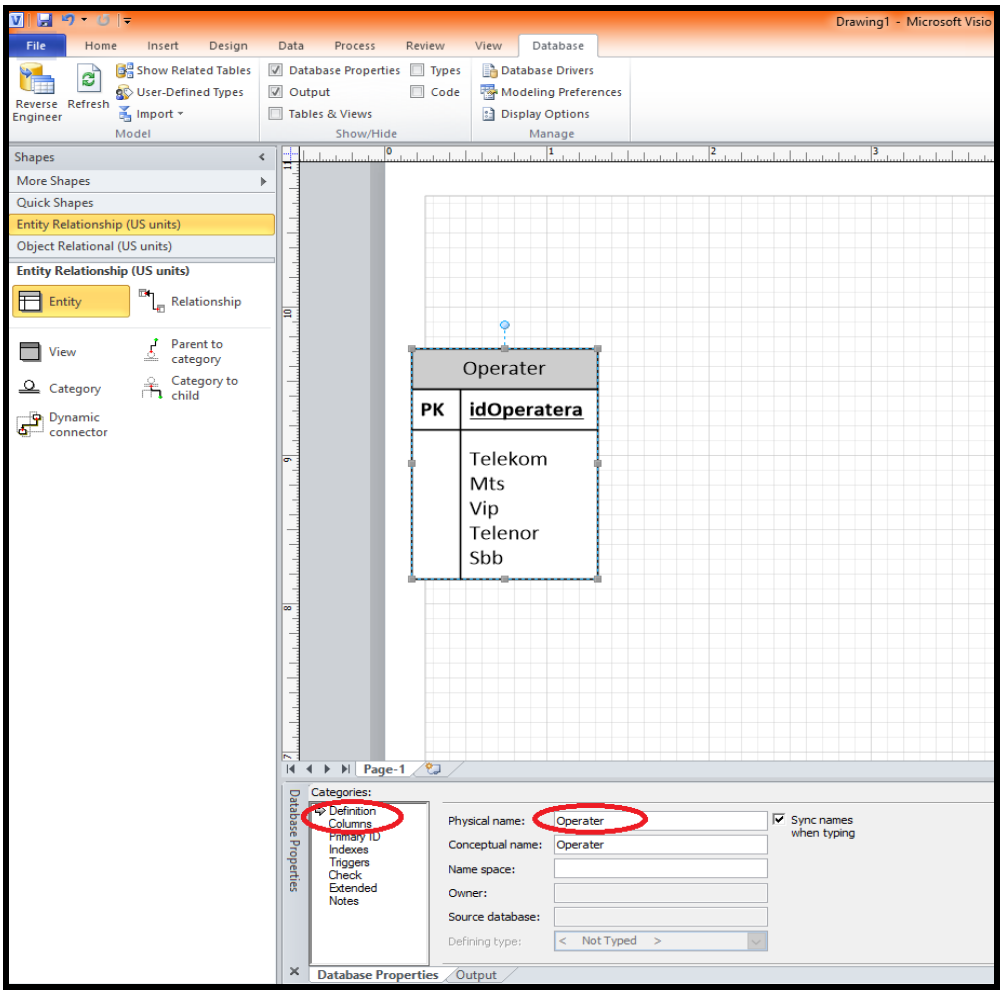
Слика 52. Приказ табеле ентитета у „Visio“ пакету

Уколико при отварању Visio пакета прозор не садржи „Database Properties“, он се може приказати чекирањем опције „Database Properties“ у главном менију „Database“, што је приказано на слици 53.



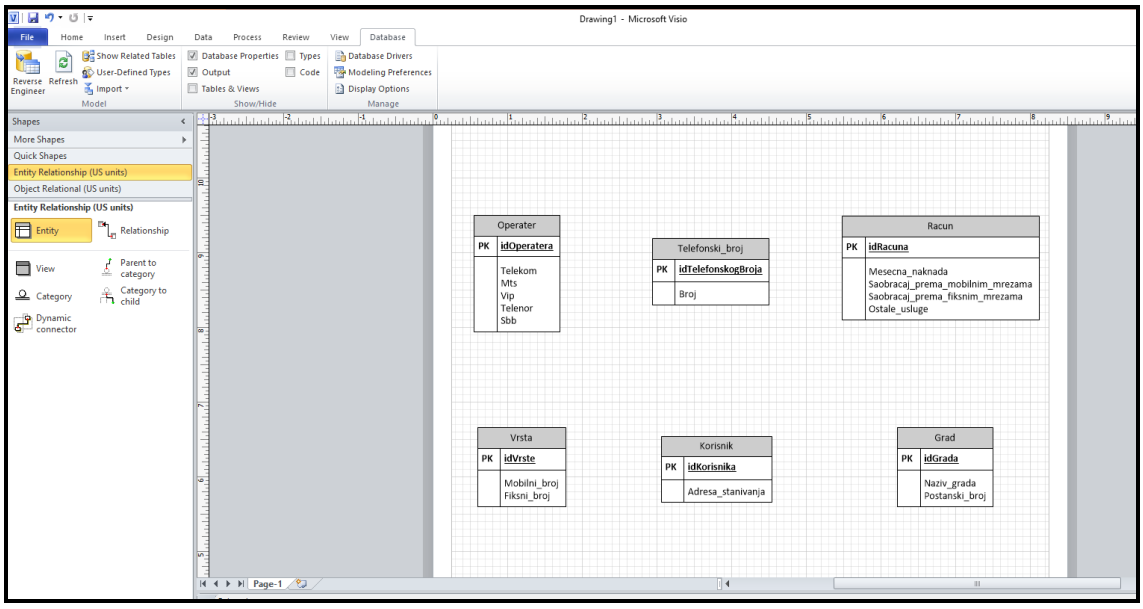
Слика 53. Приказ „Database“ таба у „Visio“ пакету

У трећем поглављу је већ споменуто да се у прозору „Database Properies“, одељку „Definition->Physical name“даје назив табели- ентитету, као и да се у одељку „Columns“ додељују атрибути. Ентитет по дијаграму са слике 18, у Visio пакету је приказан на слици 54. са својим атрибутима.



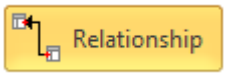
Слика 54. Приказ ентитета „Operater“ са атрибутима у „Visio“ пакету

Овај поступак треба поновити за сваки ентитет, па ће наш модел изгледати као на слици 55.



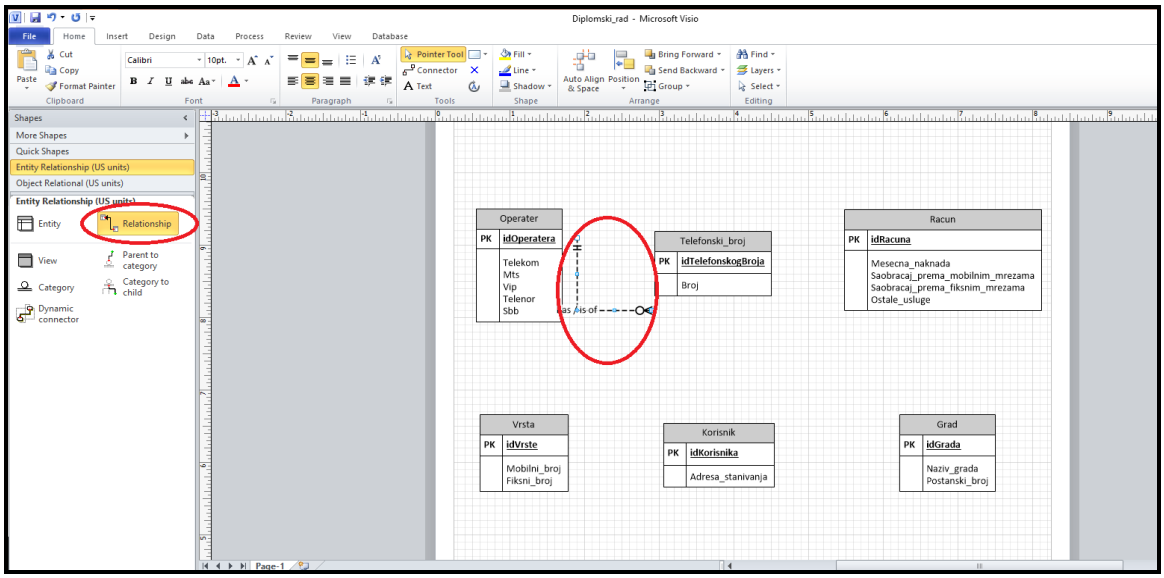
Слика 55. Приказ ентитета са атрибутима у „Visio“ пакету

Након убацивања ентитета и атрибута потребно је извршити њихово повезивање, везе се креирају превлачењем иконице „Relationship“(Слика 56.) која се налази са леве стране (прошлом поглављу званим другим, левим делом Visio прозора) у прозору под називом „Shapes“ на наш модел са ентитетима.



Слика 56. Приказ иконе „Relationship“ у „Visio“ пакету

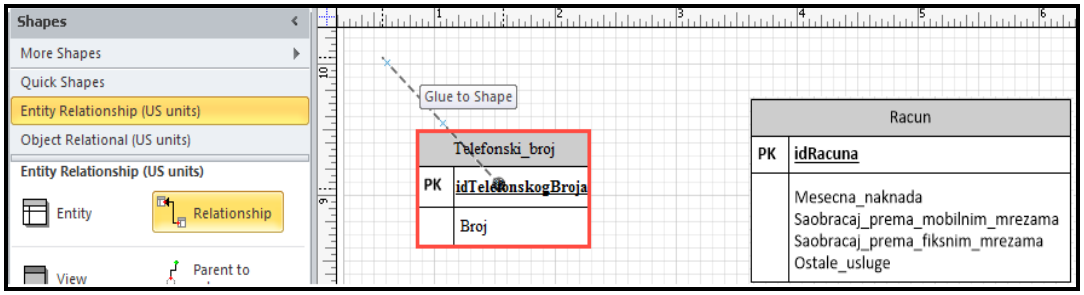
Након овог поступка у нашем моделу базе добијамо не конектовану везу као што је приказана на слици 57.



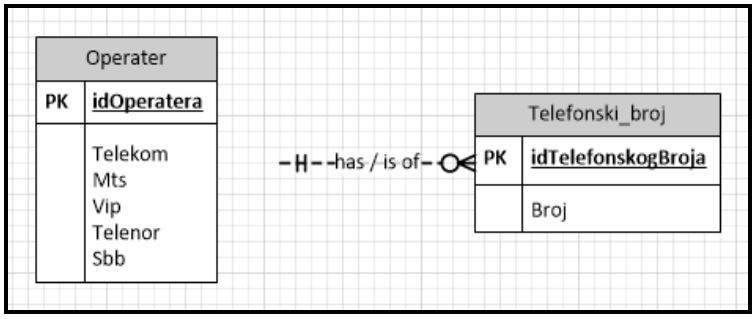
Слика 57. Приказ не конектоване везе у „Visio“ пакету

Како би веза била успостављена између табала (ентитета) потребно је превући, левим кликом миша, један крај везе на одговарајућу табелу, притом док држите крај

везе потребно је да померате миша по тој табели док цела табела не добије црвен оквир (Слика 58), и тада пустите леви клик. Добија се веза чији је један крај везан за ентитет (Слика 59).

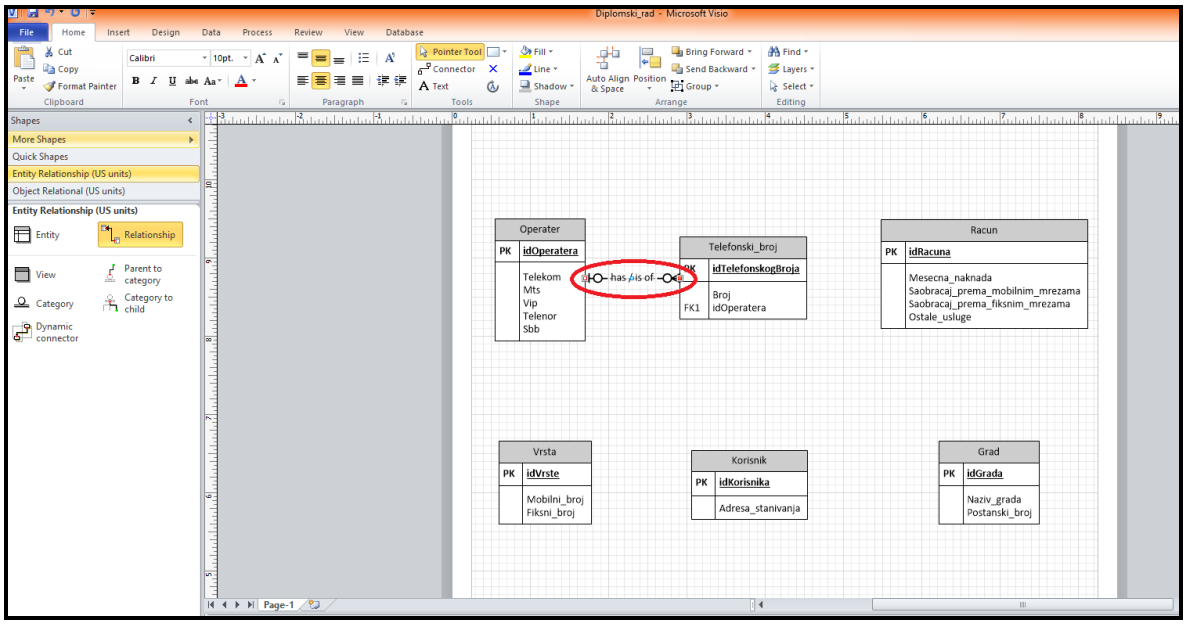


Слика 58. Приказ постављања везе и ентитета у „Visio“ пакету



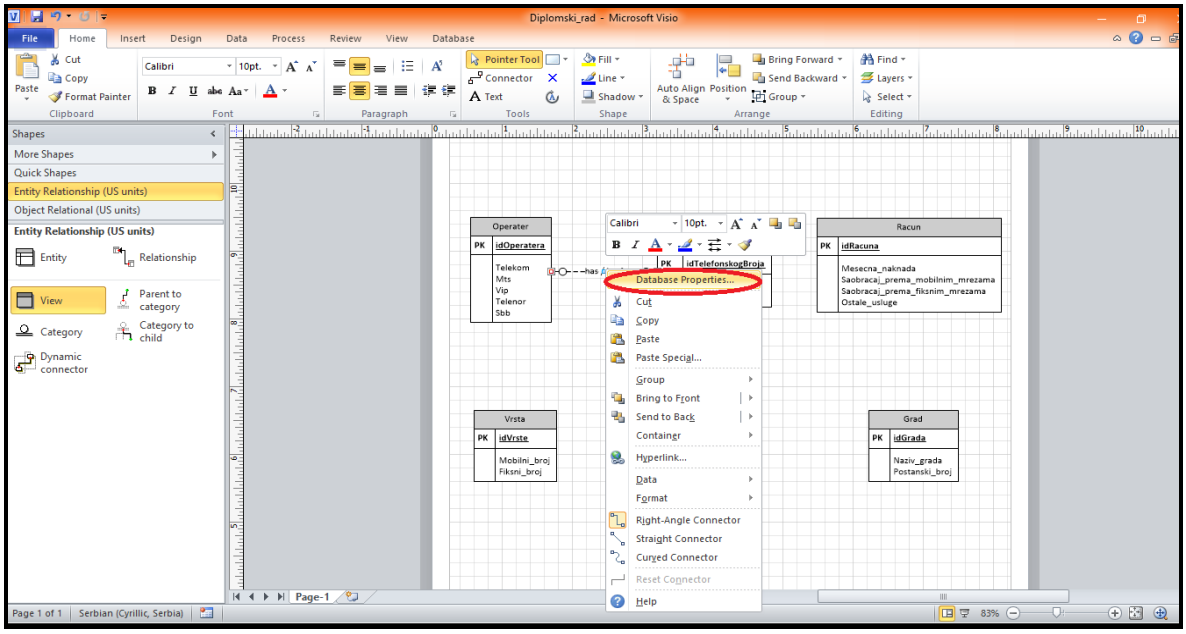
Слика 59. Приказ ентитета и везе у „Visio“ пакету

Потребно је исти поступак поновити и са другим крајем везе, тако да се добије веза између табела као што је приказана на слици 60.

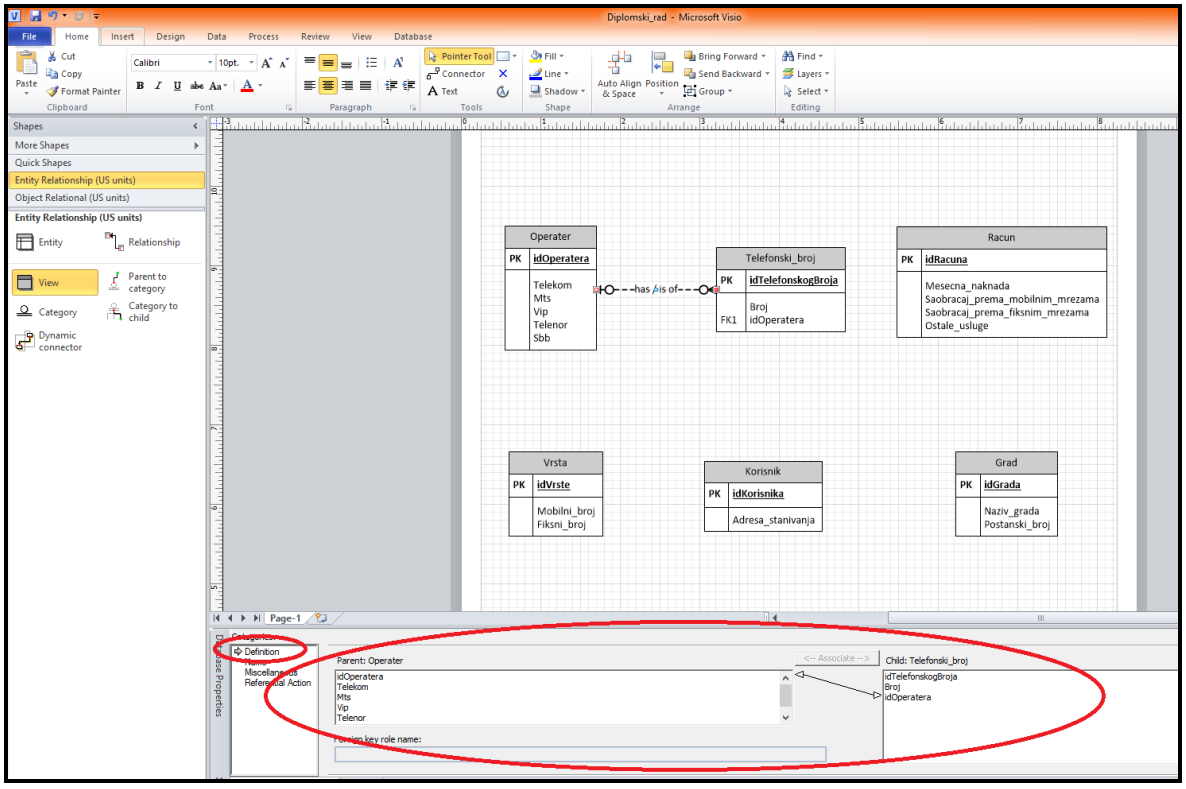


Слика 60. Приказ ентитета и везе у „Visio“ софтверу

Подешавање везе, као што је већ речено, се врши у прозору „Database Properties“, након једноставног двоструког левог клика на везу (Слика 62.) или десним кликом на везу, па кликом на „Database Properties“ (Слика 61).



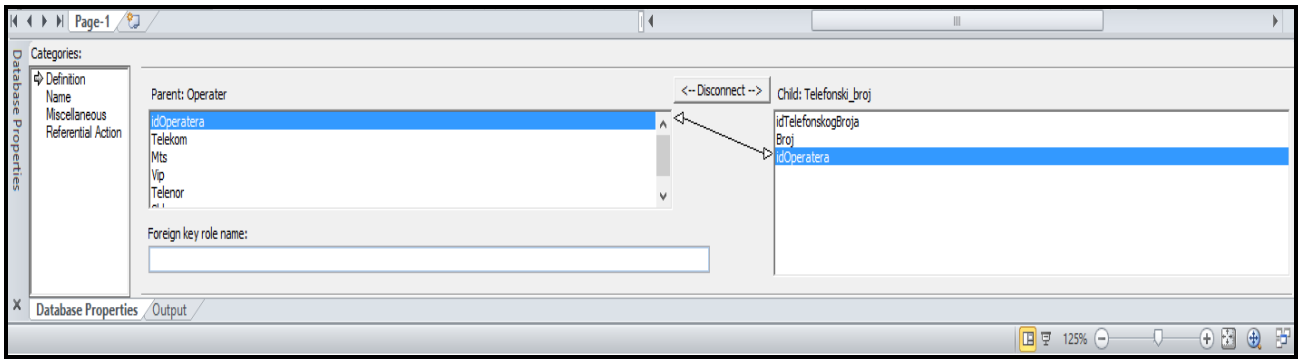
Слика 61. Приказ отварања прозора за подешавање везе („Database Properties“) у „Visio“ софтверу



Слика 62. Приказ одељка „Definition“ („Database Properties“) у „Visio“ софтверу

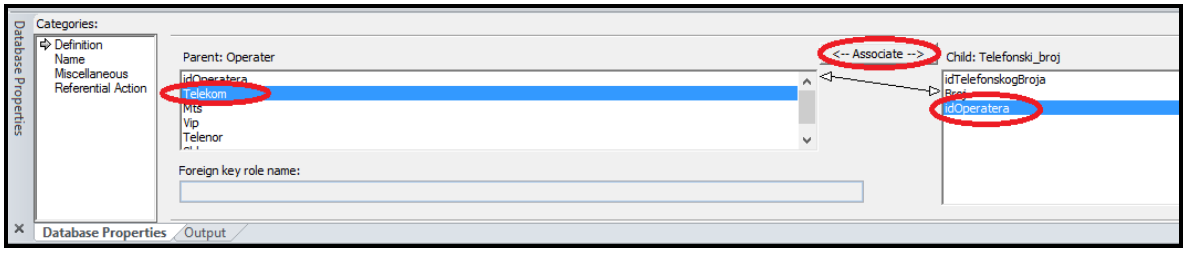
Када је отворен прозор „Detabase Properties“ , који се односи на одређену везу, његов први одељак (прва категорија) „Definition“ се односи на управљање над атрибутима ентитета, односно одређивања односа родитељ-дете („Parent-Child“), и на тај начин

стварања аутоматског спољашњег кључ у ентитету који у том одосу представља дете (Слика 63).



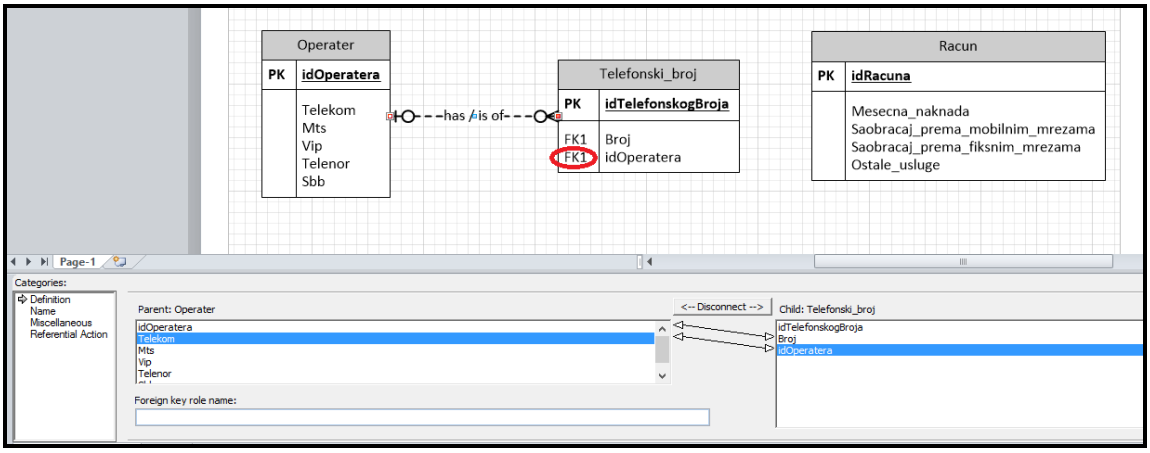
Слика 63. Приказ одељка „Definition“ („Database Properties“) у „Visio“ софтверу

Ови односи се могу врло једноставно мењати, кликом на одговарајући атрибут у „Parent“ листи, потом кликом на атрибут који се нализи у „Child“ листи, и кликом на „Associate“ дугме, врши се њихово повезивање (Слика 64).



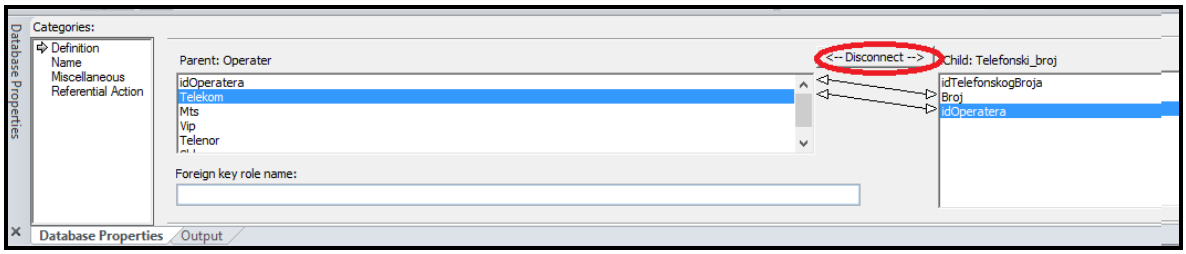
Слика 64. Приказ начина мењања односа родетељ-дете („Database Properties“) у „Visio“ софтверу

Њихово повезивање се и графички приказује стрелицама, тако да се увек може знати који су и у каквим односима одређени атрибути, што се може видети на сликама 64 и 65. У зависности од односа, спољашни кључ се аутоматски приказује у табели (Слика 65).



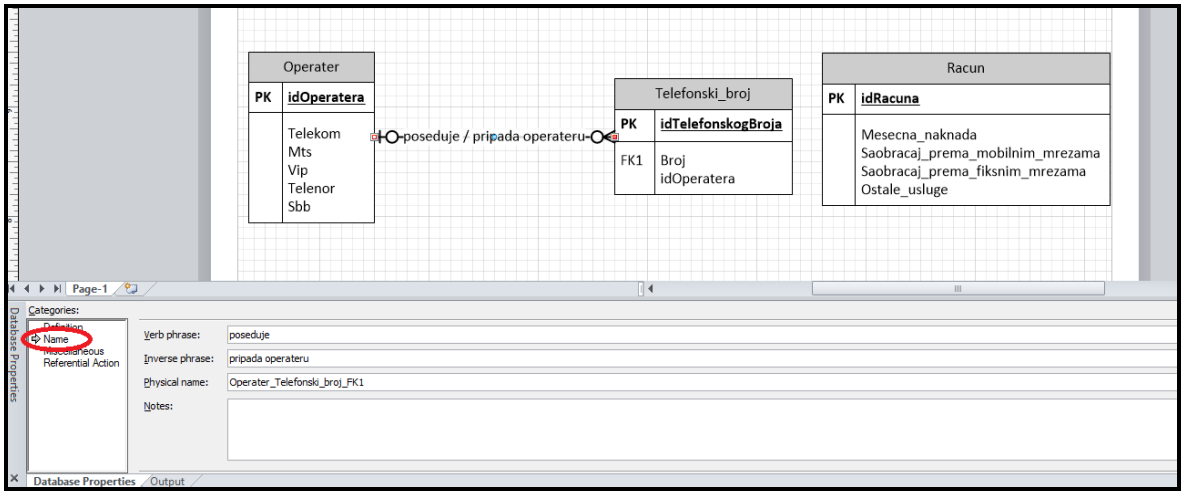
Слика 65. Приказ спољашњег кључа у табели „дете“ у „Visio“ софтверу

Када желимо да обришемо одређени однос, такође одаберемо атрибуте у обе листе и кликом на „Disconnect“ дугме (Слика 66), њихов однос је избрисан (овим поступком се бришу и стрелице које су их графички повезивале).



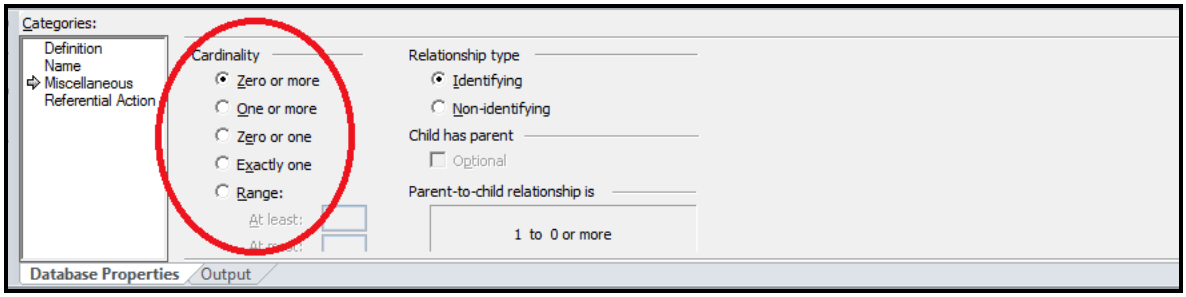
Слика 66. Приказ начина мењања односа родитељ-дете („Database Properties“) у „Visio“ софтверу

Одељак „Name“ (Слика 67.), се користи за давање (физичког) имена вези, прибележавање њених карактеристика у делу „Notes“, али и за задавање речи које би лакше објашњавале везу (кардиналност) између ентитета (и то у оба смера).



Слика 67. Приказ одељка „Name“ („Database Properties“ прозор) у „Visio“ софтверу

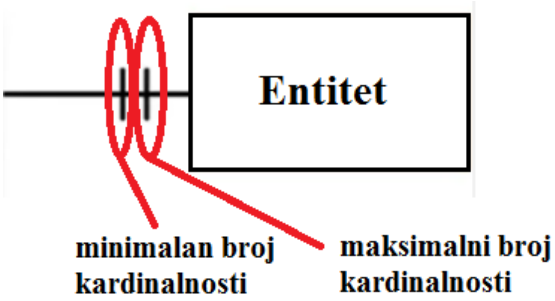
Одељак „Miscellaneous“ омогућава мењање кардиналности између ентитета, типа везе („Identifying“ и „Non-identifying“), као и опције „Optional“ и „Mandatory“ за родитеље у вези. Што се тиче кардиналности „Visio“ нуди промену одабиром и кликом испред назива кардиналности, као што је приказано на слици 68.



Слика 68. Приказ кардиналности одељка „Miscellaneos“у „Visio“ софтверу

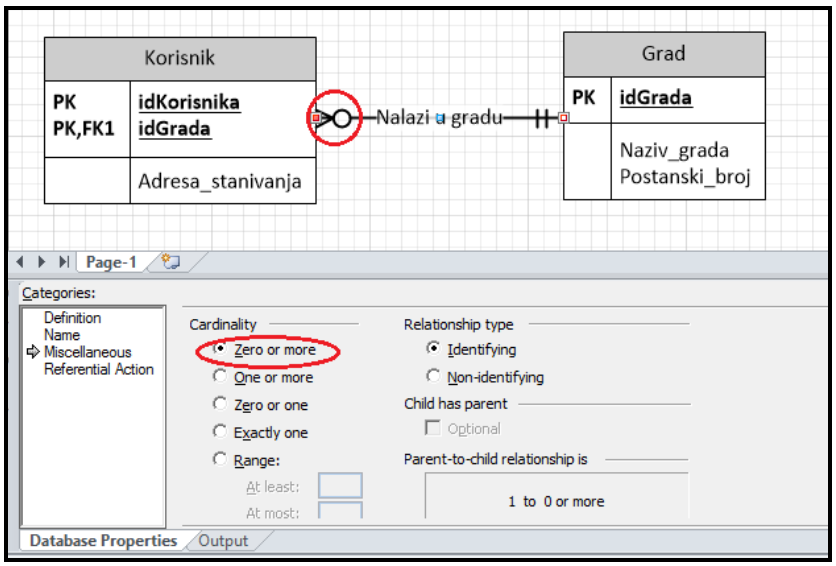
У зависности од одабира кардиналности аутоматски се мења и графички приказ везе на моделу. На графичком приказу везе уочавамо по два симбола (комбинације симбола са слике 35.) са обе стране везе, један симбол представља минималану а други максималну вредности кардиналности (односно максимални и минимални

број атрибута једног ентитета који може да учествује у неком односу - вези) (Слика 69) .



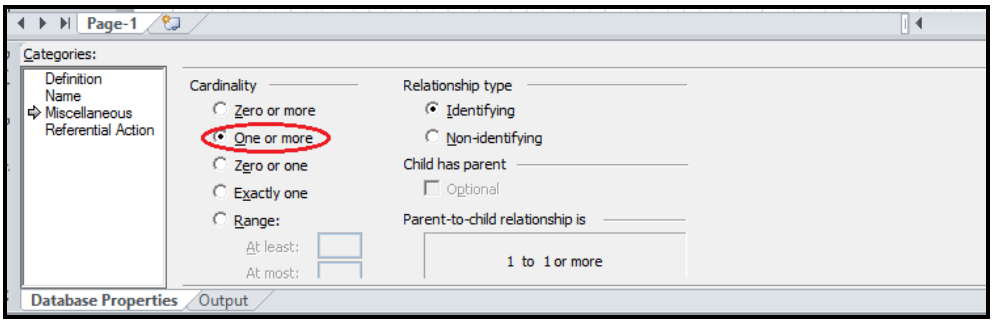
Слика 69. Приказ симбола кардиналности у „Visio“ софтверу

Када одаберемо кардиналност „Zero or more“, то подразумева да у односу „дете-родитељ“, дете према родитељу има минималан број (кардиналитета) нула, а максималан број већи од нуле, веза са ове стране има кардиналност (M,0), а у конкретном примеру значи да један град може имати више корисника, у нашем случају телефонског броја, а не мора имати ни једног, док се један корисник налази у једном граду (највише и најмање у једном граду). (Слика 70).

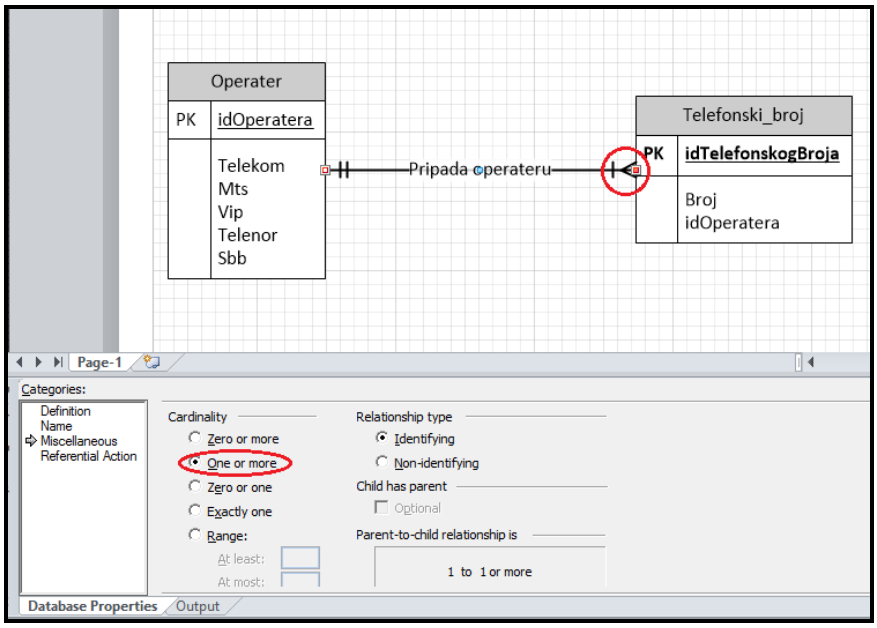


Слика 70. Приказ кардиналности „Zero or more“ са графичким симболима у „Visio“ софтверу

Кардиналност „One or more“ (Слика 71.), у односу дете-родитељ, подразумева минималан број 1 (кардиналитета) а максималан M („више“). У конкретном примеру са слике 72. значи да једном оператеру може да припада више телефонских бројева (а најмање један да би био оператер), док један тел.број може да припада само (највише и најмање) једном оператеру.

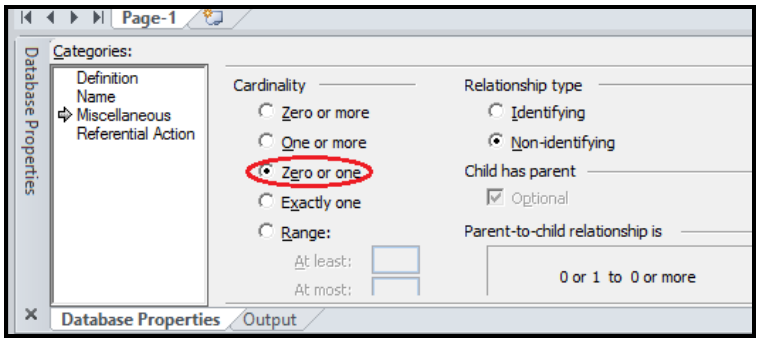


Слика 71. Приказ кардиналности „One or more“у „Visio“ софтверу

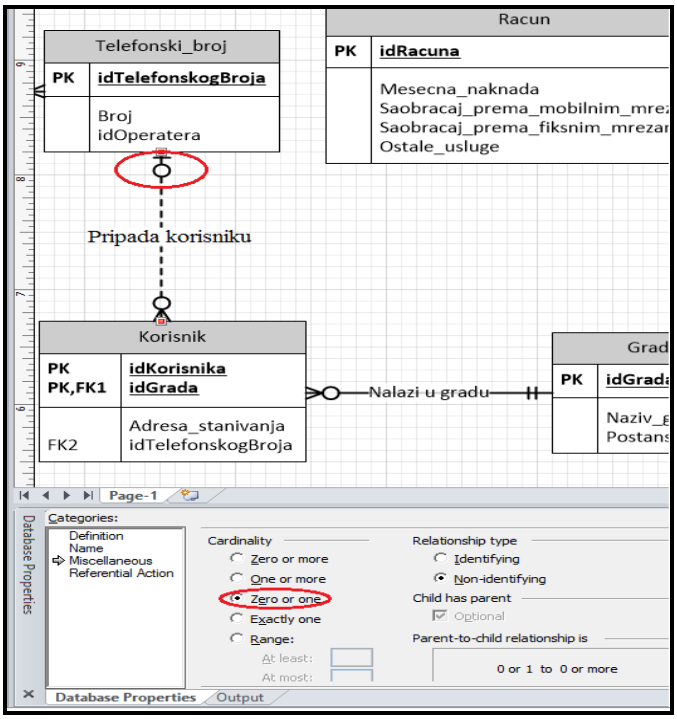


Слика 72. Приказ кардиналности „One or more“ са графичким симболима у „Visio“ софтверу

Кардиналност „Zero or one“ (Слика 73.), у односу дете-родитељ, подразумева минималан број 0 (кардиналитета) а максималан 1. Односно према слици 74. значи да један телефонски број може да има више корисника (а не мора ни једног), а једном кориснику може да припада највише један телефонски број (а не мора ни један).

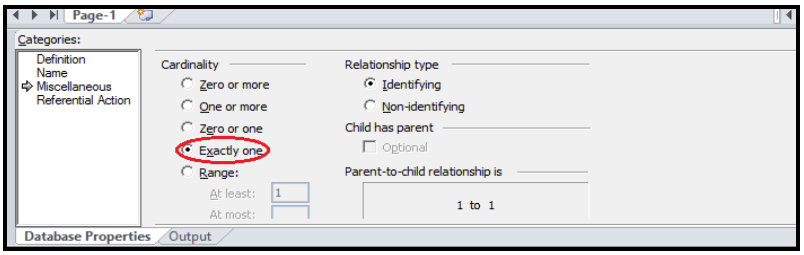


Слика 73. Приказ кардиналности „Zero or one“ у „Visio“ софтверу

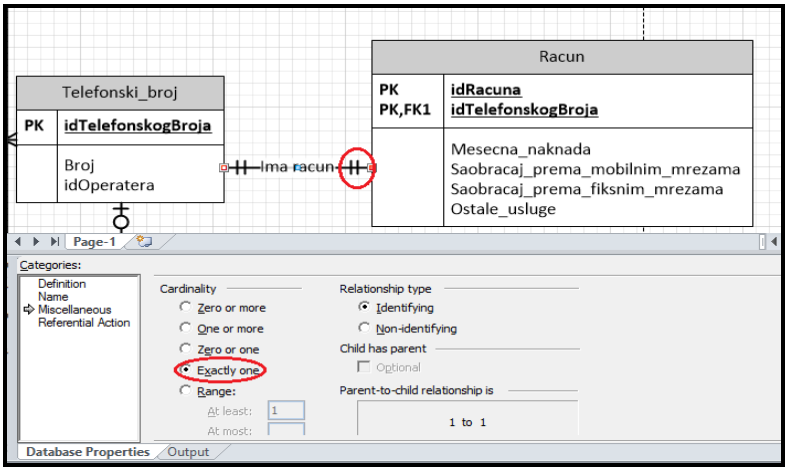


Слика 74. Приказ кардиналности „Zero or one“ са графичким симболима у „Visio“ софтверу

Кардиналност „Exactly one“ (Слика 75.), у односу дете-родитељ, подразумева минималан број 1 (кардиналитета) као и максималан 1. Односно према слици 76., један телефонски број може да поседује најмање један и највише један телефонски рачун, односно, један рачун се добија само за један телефонски број.

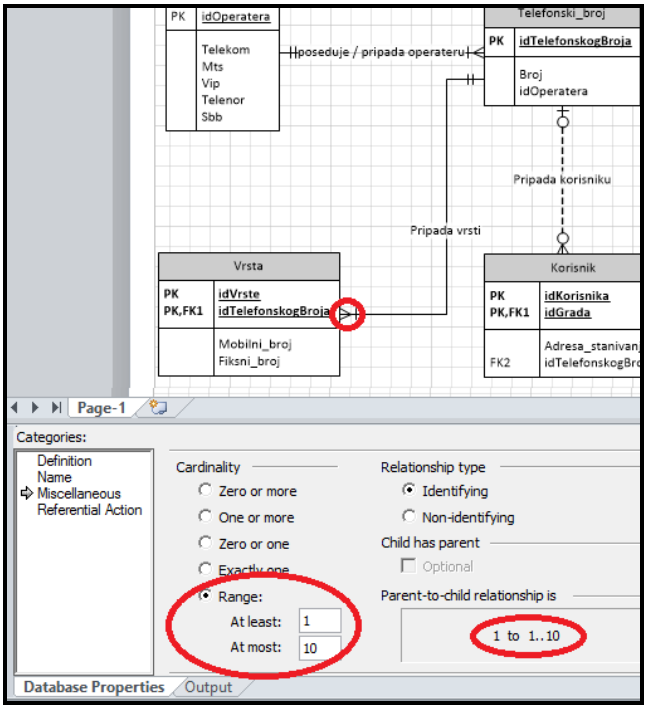


Слика 75. Приказ кардиналности „Exactly one“ у „Visio“ софтверу



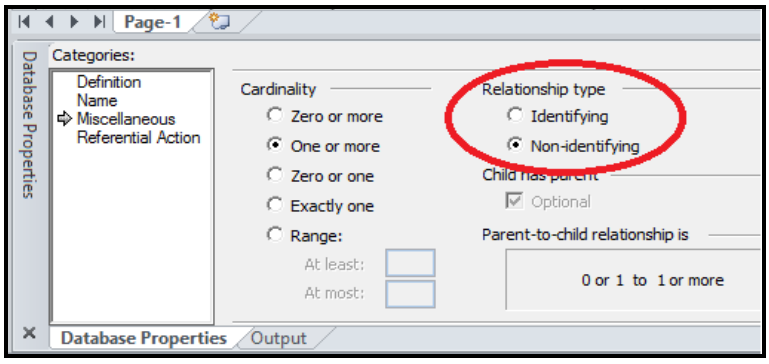
Слика 76. Приказ кардиналности „Exactly one“ са графичким симболима у „Visio“ пакету

Visio нуди опцију да сами унесемо минималне и максималне вредности кардиналности, то се може учинити одабиром опвије „Range“, што отвара два нова поља (Слика 77.), прво „At least“ што представља поље за уношење минималне вредности кардиналности и друго „At most“, што представља вредност максималне кардиналности. Графички приказ овог карданилитета се аутоматски интегрише у модел означене везе.



Слика 77. Приказ кардиналности „Range“ са графичким симболима у „Visio“ пакету

„Relationship type“ нуди две опције „Identifying“ и „Non-identifying“ (Слика 78). „Identifying relationship“, значи да дете табела не може да буде јединствено идентификовано без родитеља. То се може замислити на примеру, рецимо да се нађете у ситуацији да постоји међу табела која се користи за решавање many-to-many везе, где међу табела садржи примарни кључ који је састављен од примарних кључева леве и десне стране међу табеле (родитеља). „Non-identifying“, опција је супротна од претходне, она означава однос у коме дете може бити независно приказано (идентификовано) од родитеља.



Слика 78. Приказ дела „Relationship type“ у „Visio“ пакету

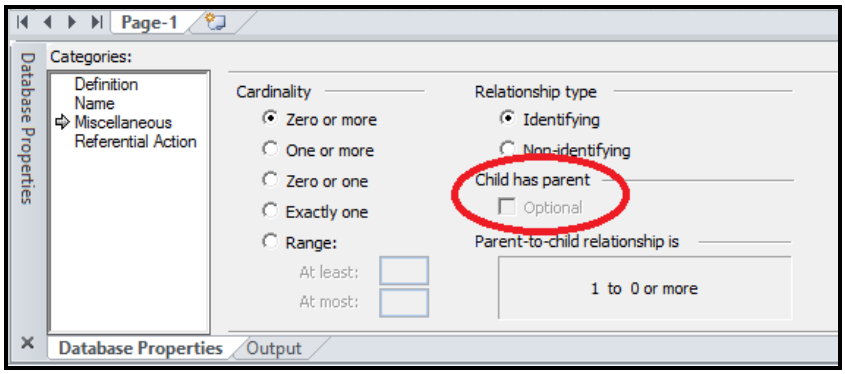
„Child has parent“, нуди опцију „Optional“(Слика 80.), уколико ова опција није чекирана подразумева се да је однос родитељ-дете „Mandatory“. На слици 79. је

приказана промена кардиналности дете-родитељ у зависности од горе наведених опција.

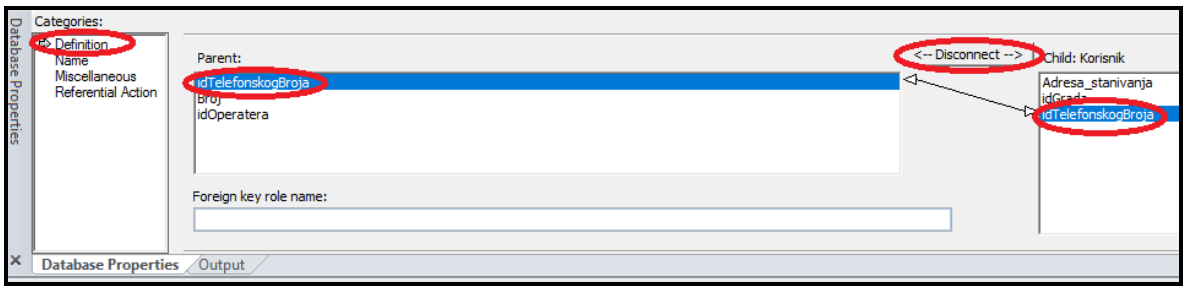
Mandatory	Optional
1 TO 0 or more	0 or 1 TO 0 or more
1 TO 1 or more	0 or 1 TO 1 or more
1 TO 0 or 1	0 or 1 TO 0 or 1
1 TO 1	0 or 1 TO 1
1 to Range [at least N, at most M]	0 or 1 to Range [at least N, at most M]

Слика 79. Приказ „Optional“и „Mandatory“ кардиналности у „Visio“ пакету

Када повезујемо табеле (ентитете) преко „relationship“ конектора некада опцију „Optional“ није могуће користити (Слика 80.), па се морају одрадити следећи кораци: брисање односа између табеле родитељ-дете у делу „Definition“(Слика 81.) и враћање у део „Miscellaneous“ како би чекирали опцију „Non-identifying“ (Слика 82).

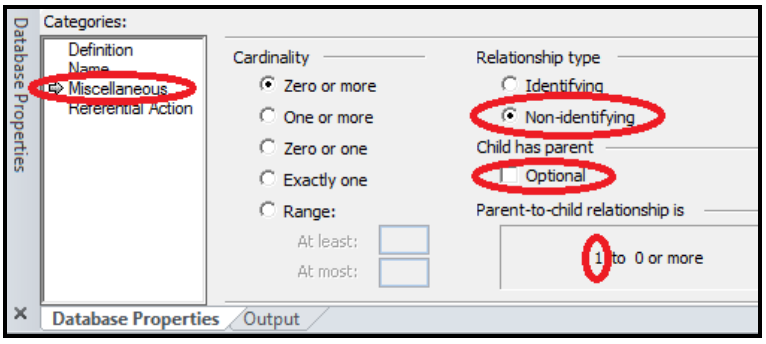


Слика 80. Приказ „Optional“ опције у „Visio“ пакету

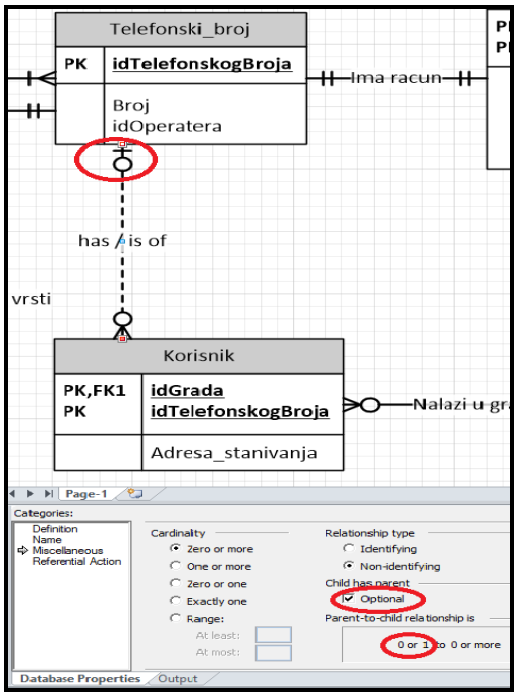


Слика 81. Приказ „Optional“ опције у „Visio“ пакету

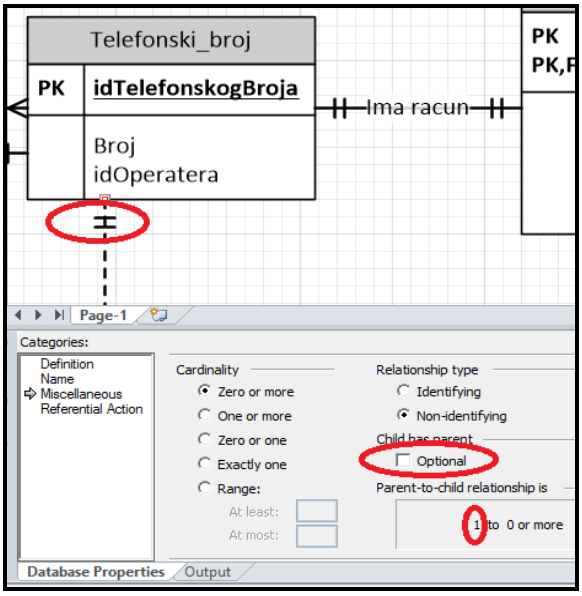
Чекирањем опције „Optional“примећујемо да се у односу родитељ-дете, од стране родитеља појављује кардиналност са минималним бројем 0 а максималним бројем 1 кардиналитета, као и гашењем исте, добија се и за минималну и за максималну кардиналност број 1. Овакву кардиналност прате и графички симболи за одабрану везу у нашем моделу, што се може видети на сликама 83 и 84.



Слика 82. Приказ отварања „Optional“ опције у „Visio“ пакету

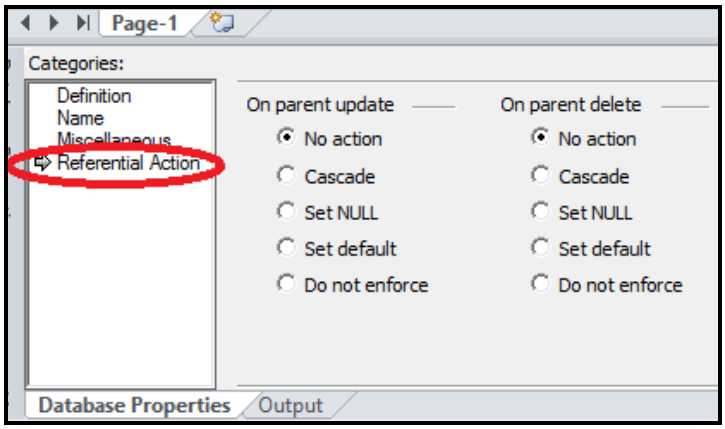


Слика 83. Приказ „Optional“ опције и промена граф. симбола у „Visio„пакету



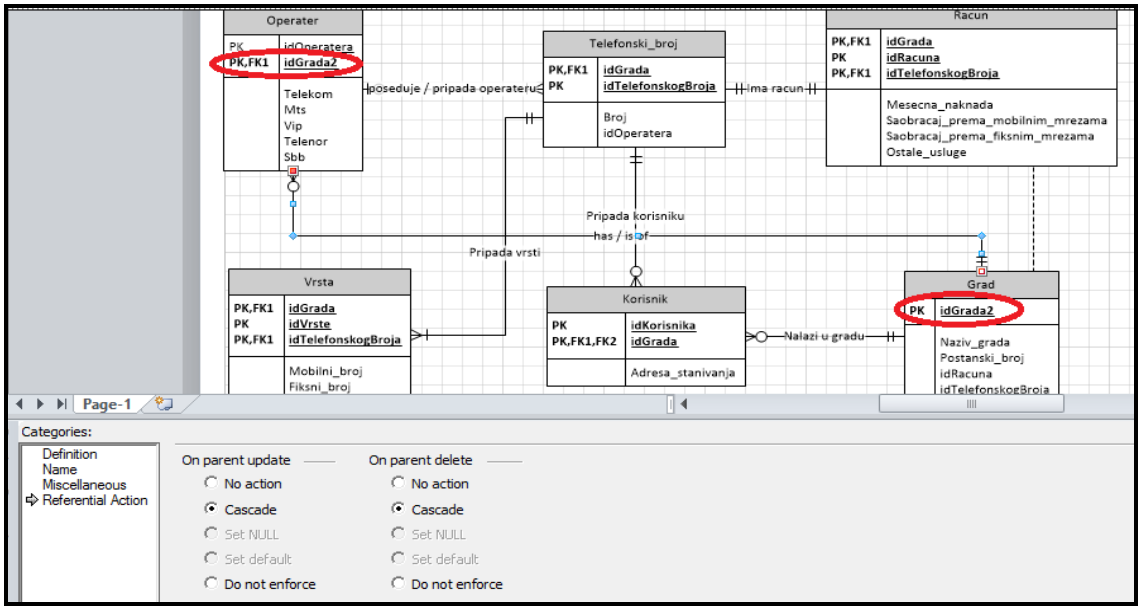
Слика 84. Приказ „Optional“ опције и промена граф. симбола у „Visio“ пакету

Последња категорија коју нуди Visio програмски пакет у прозору „Databse Properties“ (који се односи на одређену везу) је „Referential Action“, која се користи за подешавање начина функционисања дете табеле када се подаци родитељ табеле ажурирају или бришу (Слика 85).



Слика 85. Приказ категорије „Referential Action“ у „Visio“ пакету

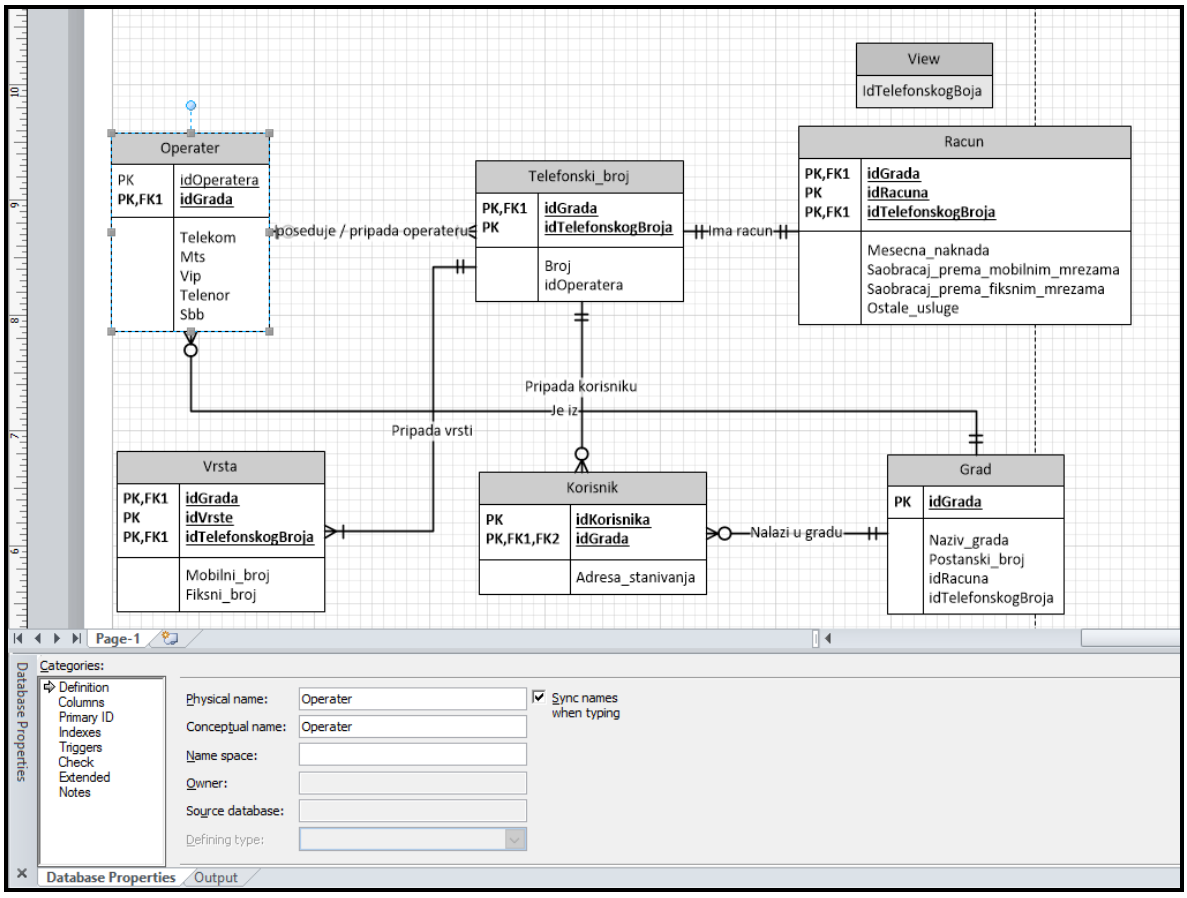
Дакле овде је могуће променити (направити изузетак) општи референтни интегритет који је постављен за целокупну базу података. Microsoft Office Visio аутоматски генерише референтна ограничења између повезаних табела, а опције које се у овој категорији промене важе искључиво за одређену везу. Опције које су понуђене за чекирање су „No action“, „Cascade“, „Set NULL“, „Set default“ и опција „Do not enforce“. Опција „No action“ подразумева гашење аутоматизоване промене у спољашњем кључу када је родитељ у изабраној вези ажуриран. Што софтвер по аутоматској шеми ради ако ова опција није чекирана, овај се поступак назива и рестрикцијом. Опција „Cascade“ омогућава да се исте промене у спољашњем кључу обаве као и у примарном, када се родитељ у изабраној вези ажурира. Примена опције „Cascade“ се може видети на слици 86. Након неке промене у примарном кључу врши се промена и у спољашњем, у примеру је извршена промена назива примарног кључа „idGrad“ у „idGrad2“, дакле захваљујући овој опцији се променио и назив спољашњег кључа у табели „Operater“.



Слика 86. Приказ опције„Cascade“ у „Visio“ пакету

Опција „Set NULL“ даје могућност сетовања спољашњег кључа на NULL вредност, када се родитељ табела у одабраној вези ажурира. Укључивањем опције „Set default“ добијамо могућност да се спољашњи кључ сетује на одговарајућу вредност, када се родитељ табела у одабраној вези ажурира. Укључивањем опције „Do not enforce“ , се дозвољава да се онемогући референцијални интегритет, када се родитељ табела у одабраној вези ажурира. Као што се види на слици 85. други део ове категорије чини „On parent delete“ форма, његове опције имају исте називе као и претходна форма, али имају другу сврху, Опција „No action“ подразумева гашење аутоматизоване промене у спољашњем кључу када се родитељ у изабраној вези брише. Опција „Cascade“ омогућава да се исте промене у спољашњем кључу обаве као и у примарном, када се родитељ у изабраној вези брише. Опција „Set NULL“ даје могућност сетовања спољашњег кључа на NULL вредност, када се родитељ табела у одабраној вези брише. Опцијом „Set default“ добијамо могућност да се спољашњи кључ сетује на одговарајућу вредност, када се родитељ табела у одабраној вези брише. Укључивањем опције „Do not enforce“ , се дозвољава да се онемогући референцијални интегритет, када се родитељ у одабраној вези брише.

На основу свих предходних корака ER дијаграм са слике 18, у Visio програмском пакету изгледа као на слици 86.



Слика 87. Приказ готовог модела у „Visio“ пакету

5. Закључак

Развој база података је у сталном напретку, некада је било незамисливо чување слика, графикона или видео записа. За базу података је био потребан меморијски уређај великих димензија, данас је то могуће на сваком хард диску (величине достижу и до неколико терабајта), постоји могућност да се база података налази и на серверу неке компаније која се бави интернет сервисима, којој се може приступити са било ког места „online“, што још више олакшава домен ажурирањања, интеграција информација, опоравка од грешке и клијент-сервер приступ, са разлогом су базе података постале део свакодневнице. Тај њихов значај је и условио напредак софтвера за њихову реализацију (прављење, моделирање, управљање). Mocrosoft Visio софтвер представља јединствен начин за пројектовање модела базе података, на лако сватљив начин почетник може да направи модел базе података и извезе га у одговарајући софтвер, а са друге стране професионалци имају широк спектар функција за управљање и креирање модела база. Visio притом настоји да његово радно окружење буде још „комфорније“за пројектанте, што се и може закључити на основу новијих верзија овог софтвера. Оне подржавају и „тимски рад“ у смислу да на једном моделу истовремено може да ради више пројектаната засебно, а притом се свака измена бележи, као и информација везана за њу, попут времена или који је пројектант учествовао у томе и слично. Microsoft компанија је имала идеју да направи универзални софтвер за потребе моделирања, документовања, прављења дијаграма и графикона у организацији, 3D пројектовању, базама као и много другим областима. Више је него очигледно да је успела у томе.

6. Литература

[1]- Павловић-Лажетић Г.: Основе релационих база података, Математички факултет, Београд, 2000.

[2]- Небојша Лукић: Базе података, Техничка школа Михајло Пупин, Бијељина. 2007.

[3]- Тонћи Дадић: Базе података, Факултет природословно-математичких знаности, Сплит, 2012

[4]- Bonnie Biafore: Visio 2007 Bible, Wiley Publishing, Inc., Indianapolis, Indiana, 2007.

[5]- Scott A. Helmers: Microsoft Visio 2010 Step by Step, Microsoft Corporation, Sebastopol, California, 2011