Факултет инжењерских наука Универзитета у Крагујевцу

Назив студијског програма: Машинско инжењерство

Ниво студија: Основне академске студије

Модул: Информатика у инжењерству

Предмет: Базе података

Број индекса: 90/2008

Милош Б. Урошевић

Развој база података применом CASE алата

завршни рад

Комисија за преглед и одбрану:

1. **Др Милан Ерић- ментор**
2. **_____**
3. **_____**

Датум одбране:_____

Оцена:_____

Изјава о самосталној изради рада

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

_____

У оквиру овог завршног рада кандидат треба да:

Проучи литературу, изврши систематизацију проученог материјала и презентира тему "Развој база података применом CASE (Computer Aidid Software Engineering) алата". Целине које треба да садржи завршни рад су: уводна разматрања везана за развој база података, примена CASE алата у пројектовању база података, упоредна анализа CASE алата, студијски пример и закључна разматрања.

Препоручена литература:

1. Лазаревић Б., и др.: **Базе података**, ФОН Београд, Београд 2003.
2. Павловић-Лажетић Г.: **Основе релационих база података**, Математички факултет, Београд, 2000.
3. Rainer K., Turban E.: **Introduction to Information Systems**, John Wiley & Sons, Inc., 2009
4. Полишчук Ј.: **Пројектовање информационих система**, Електротехнички факултет, Подгорица, 2007.
5. Вељовић А.: **Пројектовање информационих система**, Компјутер библиотека, 2003.
6. Вељовић А.: **Развој информационих система и базе података**, Компјутер библиотека, 2003.

Крагујевац, датум

ментор:

Име, презиме и звање ментора

_____

Резиме: У овим раду су презентоване теориске основе база података и Case алата, као и могућности и значај Case алата у пројектовању база података.

Кључне речи: базе података, Case алати.

Summary of work: This paper presents the theoretical foundations databases and Case tools as well as the possibilities and importance of Case tools in the design of databases.

Keywords: database, Case tools

САДРЖАЈ

3.0. АНАЛИЗА ПОЈЕДИНИХ CASE АЛАТА

1.0 УВОД У БАЗЕ ПОДАТАКА

Модерне компаније и институције поседују различите електронске (рачунарске, информационе) системе које користе као подршку у процесу обраде информација, које настају како унутар самог система тако и оних који долазе споља. Такви информациони системи обезбеђују како особљу тако и спољним корисницима (купци, добављачи, агенције и сл) да приступе информацијама компаније са различитим нивоима приоритета и права приступа. Такви системи могу да буду системи за управљање докумената, системи за управљање пројектима, e-mail системи, интернет странице и сл. Такви системи имају један неизоставан део - систем база података, која чува све информације које се обрађују и обезбеђује приступ тим информацијама. Базе података су кључна компонента код стандардних информационих система, али и других Web заснованих апликација. Користе их орагнизације и предузећа од оних најмањих до глобалних корпорација и милиони корисника.

1.1 Систем база података

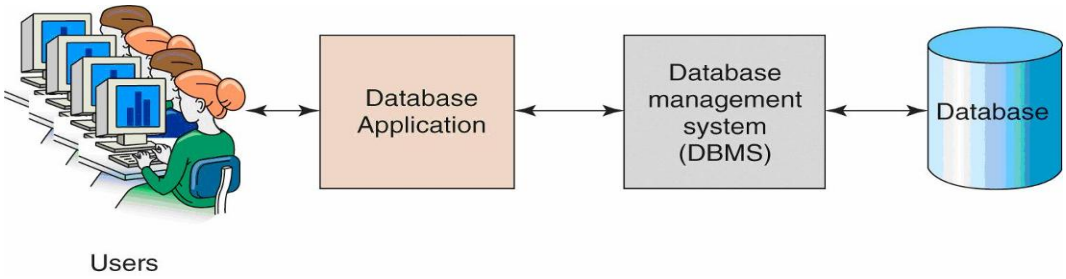
Систем база података садржи 4 основне компоненте (слика 1.)

(1) корисници,

(2) апликација над базом података,

(3) систем за управљање базама података (Database Management System - DBMS), и

(4) база података.



слика 1. Компоненте система база података

1.2. Базе података

База података представља колекцију међусобно повезаних података који су организовани у табеле и друге структуре података, а користе за једну или више апликација.

Основна намена базе података је да буде **репозиторијум** (складиште) за податке. Подаци могу бити различитог типа, текстуални, нумерички, слике, аудио и видео записи и сл.

Из „дефиниције" базе података види се да је она **колекција међусобно повезаних података организованих у табеле**. У овој „дефиницији" две су чињенице од значаја - организација података у табеле и њихова међусобна повезаност.

Подаци у базама података су организовани у **дводимензионалне табеле**. Табела може да има више **колона**, где свака колона представља неку особину или атрибут. Врсте табеле чине конкретни подаци, односно конкренте вредности особина/атрибута неког објекта.

На пример, једна табела може да садржи информације о ученицима. Колоне табеле могу да дефинишу име, презиме, годину рођења ученика, и сл. Врсте у таквој табели су ученици, тако да се свака врста односи на једног ученика.

Које ће табеле да садржи база података зависи од проблема за који треба реализовати базу података. На пример, база података се може одности на школу, па ће у том случају табеле бити о ученицима, наставницима, одељењима, и сл. Поступак избора и дефинисања табела за базу података је део процеса моделирања односно изградње **модела података**.

Међусобна повезаност података је оно по чему се база података разликује у односу на фајл системе (датотеке) и програме за унакрсна израчунавања ко што је Ехцел. Повезаност података обезбеђује значајне предности код претраживања када корисник може да на основу веза извуче много више података. На пример, ако постоји табела која чува податке о ученицима и табела са подацима о одељењима, веза између ученика и одељења може да обезбеди да одговарајућим захтевом (SQL упитом) извучете све ученике жељеног одељења.

База података садржи и тзв. **метаподатке**, односно податке о самој структури базе података. Метаподаци могу да се односе на имена табела, имена колона у свакој табели, на податке о корисницима података, као и разним помоћним структурама које обезбеђују брз прступ подацима (индекси).

1.3 Типови програма за рад са базама података

За рад са базама података се генерално користе две врсте софтверских апликација: програми зауправљање датотекама (file management programs) и системи за управљање базама података (data base management systems – **DBMS**)

1.3.1. Систем за управљање базама података (DBMS)

Софтверски систем који омогућава корисницима дефинисање, ажурирање и контролу приступа бази података назива се систем за управљање базама података (енг. *Database Management System - DBMS*). *DBMS* обично нуди:

- **Језик за опис података** (енг. *Definition Language - DDL*). DDL, који омогућава корисницима дефинисање типа и структуре података, као и ограничења над подацима меморисаним у бази података (CREATE TABLE наредба).
- **Језик за манипулацију подацима** (енг. *Data Manipulation Language - DML*), који омогућава корисницима уметање, ажурирање, брисање и претраживање података из базе података (*SELECT, INSERT INTO, UPDATE* наредбе).
- **Језик за дефинисање начина меморисања података** (енг. *Storage Definition Language - SDL*), који се користи за специфицирање интерне шеме базе података.
- **Контролисани приступ бази података**, што укључује различите функције и механизме за приступ подацима у бази података

1.3.2. Функције DBMSa

DBMS треба да обезбеди следеће функције за контролисани приступ подацима у бази података:

- Сигурносни систем, који онемогућава приступ бази података неауторизованим корисницима (сигурносни сервиси), односно само ауторизовани корисници могу да користе податке у складу са дефинисаним привилегијама (ауторизациони сервиси)
- Интегритетни систем, који одржава конзистентност података у бази података, односно да се све промене дешавају у складу са дефинисаним правилима.

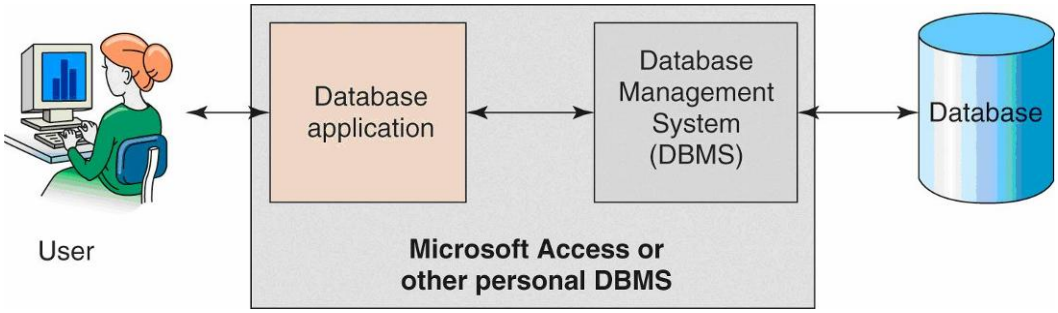
- Систем за контролу конкуренције, који допушта дељиви приступ подацима из базе података, тј да се обезбеди коректно ажурирање података када више корисника покушава истовремено да врши ажурирања.
- Систем за контролу опоравка базе података, који омогућава реконструкцију претходног конзистентног стања у случају неке хардверске или софтверске неисправности.
- Каталог коме корисници могу приступати, који садржи опис података који су меморисани у бази података.
- Подршка за трансакције, која обезбеђује коректно извршавање низа трансакција које могу бити међусобно зависне; трансакција је скуп операција уписа и читања из базе података који се третира као целина тј има свој почетак и крај.
- Разне корисничке функције, као што су импорт, експорт података, статистичке анализе, функције за надгледање,...

1.4. Типови система база података

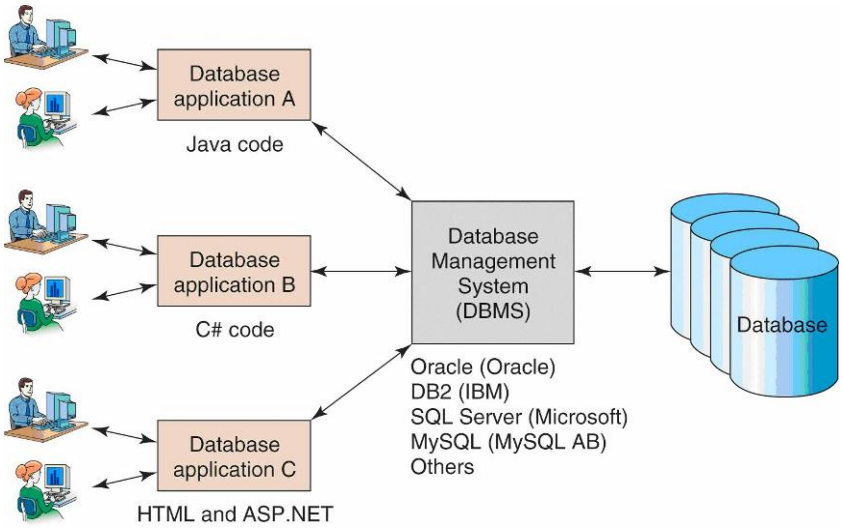
Технологија база података се може користити за велики број апликација. Практично данас скоро и да не можете да реализујете апликацију која не користи неки систем база података за чување података - без обзира да ли се ради о стандардним десктоп апликацијама, као што су књиговодствене апликације, системи за управљање документима, системи за банке, и сл, или се ради о модерним Web апликацијама које обезбеђују сложену функционалност у дистрибуираном окружењу, од он-лине куповине до разних социјалних мрежа и сл.

Један гранични случај је да вам треба апликација за евиденцију кућних трошкова. У том случају она обично садржи само неколико табела, где свака табела може да има само неколико стотина врсти. Апликацију, а самим тим и базу података, користите само ви, односно само један корисник. За такве системе се обично користи назив **персонални системи база података** (слика 2.). Наравно, овакви системи могу да се примене и на много сложеније апликације од евиденције кућног буџета. На пример, могу да покрију и пословане мањег предузећа, или да подрже рад неког WEB сајта.

С друге стране, ако имате велику компанију која има више организационих јединица, где свака од њих има сопствене пословне процесе, неопходна вам је подршка система база података који може да обезбеди чување и претрагу велике количине информација на више дистрибуираних локација. Такви системи садрже велики број табела, а неке од њих могу да имају и неколико стотина хиљада врста и више. Подацима може конкурентно да приступа велики број корисника. Такви системи обично морају да раде 24 часа дневно, 7 дана у недељу. Такви системи су познати као ***enterprise*** системи база података (слика 3.)



слика 2. Персонални систем база података



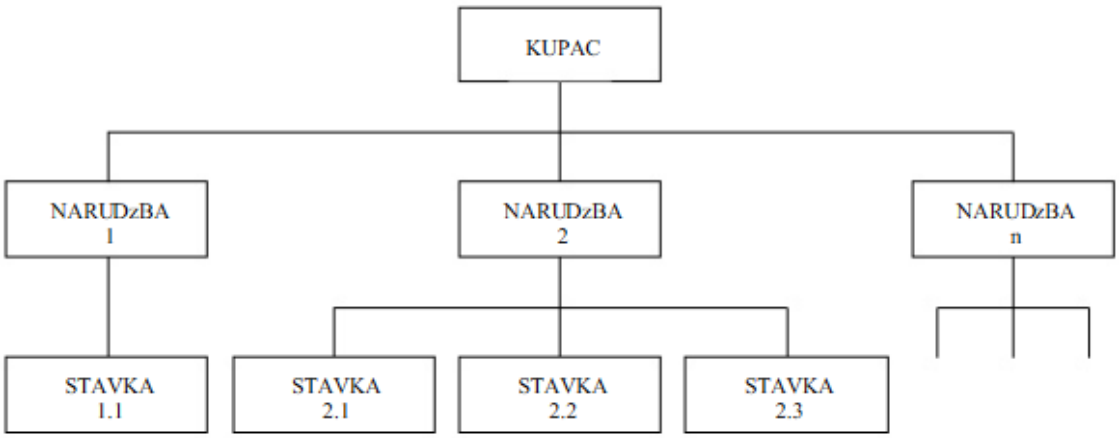
слика 3. *Enterprise* систем база података

1.5 Врсте база података

Постоје различите врсте база података, зависно од тога, на који начин су подаци интерно организовани. Тако се разликују:

- Хијерархиски модел
- Мрежни модел
- Релациони модел
- Објективни модел

Хијерархиски модел представља настарие решење у подручју базе података. основна предност се огледа у томе што је модел изузетно прегледан, до сваког податка, на било којем хијерархиском нивоу долази се само само једним приступним путем (Аццесс Патх), креће се увек од податка највишег логичког реда према низем хијерархиском ранговима. Овакав начин претараживања хијерархиске базе података омогућен је због тога што се подаци физички повезани један са другим серијом показиваца адреса, који обликују ланац повезаних ентитета података. (слика 4)

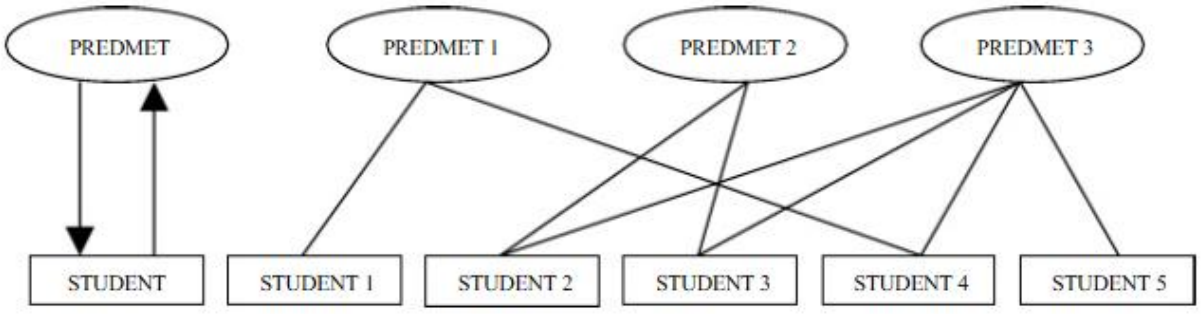


слика 4. Пример хијерархиског модела

Овај модел има један недостатак који се сматра највећим његовим недостатком. Управо због само једног приступног пута до података, апликативни програми различитих врста и намена морају се прилагођавати свакој појединој датотеци у бази података што се назива проблем координације апликативних програма са базом података.

Мрежни модел развијао се из хијерархиског, заснива се на мрежи података повезаних тако да не постоји ни основни ни подређени сегменти. У мрежним моделима и

базама података, бројним приступни путеви се могу вишеструко укрстати и гранати, што у екстремним случајевима може изазвати проблеме новог типа. Иако се мрежном структуром података смањује редундантност података и скраћује бреме одзива система приликом тражења података, превелик број могућих приступних путева превише компликује систем, па он настаје спор и неефикасан (слика 5).



слика 5. Пример мрежног модела

Релациони модел се примјењује кад због отежаности веза не могу да се употребе хијерархиски и мрезни модел. У таквим случајевима, се подаци се представљају у виду дводимензионалини табела. Рационални модел представља скуп повезаних датотека, која податке представља у облику једноставних дводимензионаланих табела које се повезују релацијама. Кад се подаци сместе у табеле, могуће је с њима обављати разне операције нпр. приказивање, додавање, брисање, претраживање и штампање.

Објектни модел је инспирисан објектно-орјентисаним програмском језицима. База је скуп трајно похрањених објеката који се састоје од својих интерних података и операција за руковођење са тим подацима. Сваки објекат припада некој класи. Између класа се успостављају везе наслеђивања, односно међусобно коришћења операција.

1.6. Релациони модел података

Релациони модел је свакако најпопуларнији и најраспрострањенији модел података данас и представља основу за релационе базе података које доминирају на тржишту. Релационе базе података доминирају на тржишту већ скоро 40 година!

Систем P је први систем који је користио релациони модел, након тога IBM је имплементирао свој систем познат као DB2. Након тога је Oracle реализовао свој систем заснован на овом моделу,... и све остало је историја.

Релација, као основни концепт релационог модела је заправо математичка релација, и има једноставну репрезентацију у облику табеле са подацима.

1.6.1. Основе релационог модела

Релациони модел има снажну теоријску основу, која се заснива на математичкој теорији релација и на логици првог реда, и за корисника врло прихватљиву репрезентацију у виду дводимензионалне табеле.

У релационом моделу података релације се користе за чување информације о објектима које треба представити у бази података. У фази пројектовања базе података, за конкретан проблем, треба најпре препознати објекте реалног света (ентитети) за које треба чувати податке и препознати њихове атрибуте. Сваки такав објекат представља се релацијом у релационом моделу.

Атрибути су заједничке особине које поседују сви ентитети једног скупа ентитета. Из скупа атрибута ентитета за потребе конкретног информационог система бира се само одређени подскуп. Пошто се ентитети односно објекти реалног света представљају релацијом, атрибути представљају својства те релације. Пошто је релација представљена табелом, атрибути представљају колоне те табеле. Свака релација представља скуп торки, где се свака торка односи на конкретан ентитет из скупа ентитета. У табеларном приказу релације, врсте табеле су подаци о конкретним ентитетима, односно торке, тако да атрибут за сваки конкретни ентитет из скупа ентитета поседује одређену вредност.

Скуп вредности које неки атрибут може узимати зовемо **домен атрибута**. Практично, сваки атрибут у релацији је дефинисан над неким доменом. Концепт домена је врло важан. Омогућава кориснику да дефинише на једном централном месту значење и извор

вредности које атрибут може узимати. Сваки домен атрибута се дефинише: типом података, дужином података и опсегом вредности.

1.6.2 Преглед основних концепата релационог модела података

Релација се у бази података представља дводимензионалном табелом, где врсте одговарају појединим слоговима, а колоне атрибутима. Атрибути се могу појављивати у било ком редоследу у табели. Редослед врста табеле такође није битан. Свака табела, као и свака колона у табели имају име.

Основни концепти релационог модела података су:

- **Релација**

Релација одговара појму табела са врстама и колонама.

- **Атрибут релације**

Представља особину ентитета представљеног релацијом. Атрибут је практично именована колона релације односно табеле, које се односе на својства објекта представљеног релацијом.

- **Домен атрибута**

Домен је скуп дозвољених вредности за један или више атрибута. Практично се односи на тип податка за колону.

- **Торка релације**

Торка је врста релације и односи се на један слог података.

- **Степен релације**

Број атрибута релације (унарна, ако има једну колону, бинарна са две колоне и сл)

- **Кардиналност релације**

Број врста (торки) релације.

- **Шема релације**

Шема релације је опис релације. Садржи име релације, имена атрибута и домене атрибута.

- **Релациона база података**

Колекција нормализованих релација.

- **Шема релационе базе података**

Скуп шема релација, при чему свака има различито име.

Очигледно је да је релација у релационом моделу, одговара појму табела у бази података. У свету база података, обично се користе једни термини када се говори о релационом моделу, а други када се говори о бази података, односно имплементацији релационог модела. Упоредни приказ термина и њиховог значења дат је на слици 6.

Релациони модел	**База података**
Релација	Табела
n - торка	Врста
Атрибур	Колона
Домен атрибута	Тип података колоне
Шема релације	Опис табеле

слика 6. Еквивалентни скуп појмова

1.6.3. Својства релације

Релација има следећа својства:

- Свака релација има име које се разликује од имена свих осталих релација у шеми релационе базе података,
- Свака ћелија табеле (одређена врстом и колоном) којом је релација представљена садржи само једну атомичну (просту) вредност,
- Сви атрибути једне релације имају различито име,
- Све вредности једног атрибута су из истог домена,
- Све торке релације су различите, тј. у релацији не постоје дупле торке,
- Редослед атрибута у релацији нема значаја, и
- Редослед торки у релацији теоретски нема значаја, али практично редослед торки у релацији може утицати на ефикасност приступа торкама!

2.0. CASE АЛАТИ У ПРОЈЕКТОВАЊУ БАЗА ПОДАТАКА

2.1. Дефиниција CASE алата

Computer Aided Software Engineering (CASE) алати служе за аутоматизацију софтверског инжењерства и самим тим представљају основни алат којим се служи пројектант информационог система.

Прве дефиниције CASE алата су подразумевале да представљају системе чији су циљеви да дефинишу, интегришу и аутоматизују што је могуће више фаза у развоју софтвера. CASE алати омогућавају да развијање софтвера постане више инжењерска делатност, а мање индивидуална уметност и умеће Зависно од тога које фазе пројектовања и имплементације CASE алати покривају, они се деле на CASE алате на вишем и CASE алате на нижем нивоу. CASE алати на вишем нивоу покривају прве фазе у производњи софтвера (анализу система и пројектовање), а CASE алати на нижем нивоу пружају помоћ у фазама програмирања.

CASE систем представља алат који служи као помоћ пројектанту информационих система. Од ефикасности овог алата може да зависи квалитет готовог производа (информационог система), тако да је пројектанту веома важно да одабере прави алат који ће га замењивати у већини мануелних услова везаних за пројектовање.

Од ефикасности CASE алата може зависити квалитет готовог софтверског производа. Успешним коришћењем правилно одабраног CASE алата може се:

- минимизирати време и труд (коштање) развоја софтвера,
- вишеструко повећати продуктивност у писању софтвера,
- подићи ниво квалитета,
- повећати поузданост,
- стандардизовати произведени софтвер.

2.2. Поделе CASE алата

Поред наведене поделе case алата на више (UPPER) и ниже (LOWER), која је уједно и најшире прихваћена, постоје и следеће две поделе:

1. **хоризонтална - временски:**
   - виши алати - за више фазе животног циклуса (кориснички захтеви и дизајн),
   - средњи алати - за средње фазе животног циклуса (имплементација - израда),
   - нижи алати -з а ниже фазе животног циклуса (подршка експлоатацији);

2. **вертикална - по функцији:**
   - алати за управљање, планирање и процене,
   - технички алати (реализација),
   - алати за подршку појекту (складишта, речници)

Интегрисани CASE алати (Integrated CASE или I-CASE) чине алате који покривају више временских фаза и више функција. Пример таквог алата је Oracle Designer/2000, који у себи садржи читав низ алата базираних на интегрисаном речнику и који покрива све фазе израде једног информационог система.

Алати могу подржавати различите технике моделирања пословних система, па се и по томе могу поделити. Најраспростањеније технике моделирања су:

- ЕР методологија (или проширена ЕР методологија) по нотацији Chen-a, Bachman-a...;
- SSA (системска структурна анализа за моделирање функционалног аспекта посматраног система) по нотацији Yourdon/de Marco, Gane/Sarson, Ward/Mellor;
- дијаграми тока програма (flow chart);
- функционалне хијерархије (декомпозиције пословних функција);
- објектно-оријентисане методологије, као OMT и Grady Booch;
- дијаграми транзиције стања.

.

Квалитету CASE алата доприноси чињеница да могу подржавати више техника моделирања.

Алати се могу поделити и по филозофији рада са њима на:

- алате са ограничавајућом (рестриктивном) филозофијом, који највише помажу новим корисницима, јер их ограничавају на ствари које прво морају да ураде; нпр., да нацртају
  контекстни дијаграм у SSA методологији, што пружа осећај сигурности, мада, можда, мало гуши креативност;
- алате са вођеном филозофијом, који су погодни за кориснике вишег нивоа предзнања и интелектуалне задатке вишег нивоа, јер само воде корисника и асистирају му у бирању операција;
- алате са флексибилном филозофијом који допуштају потпуну слободу кориснику, чак и да генерише некоректан дизајн; погодни су за кориснике који су већ у потпуности овладали методологијом и хоће да алат прати њихов креативни процес

стварања који иде мало методом са врха - надоле, а мало методом са дна - навише. Овакви алати највише врше синтаксне провере интерно и одмах по уносу, а касније провере исправности (конзистентности) дизајна, и то искључиво на захтев корисника.

Даље гледано, CASE алати могу бити:

- једнокориснички - пројекти се воде на једном месту и само један корисник употребљава алат у једном моменту;
- вишекориснички - омогућавају комункацију и координацију великих тимова за развој софтвера; свима су доступни планови, статусни извештаји, спецификације, модификације, изворни код и тестни подаци.

Постоји још велики број начина на које се CASE алати могу класификовати (на пример, по ардверском окружењу и оперативном систему у коме раде, по језику на коме су развијани или по отворености архитектуре),

2.3 Елементи CASE алата

Репозиторијум - Активни речник података који подржава дефинисање различитих типова објеката и њихових веза

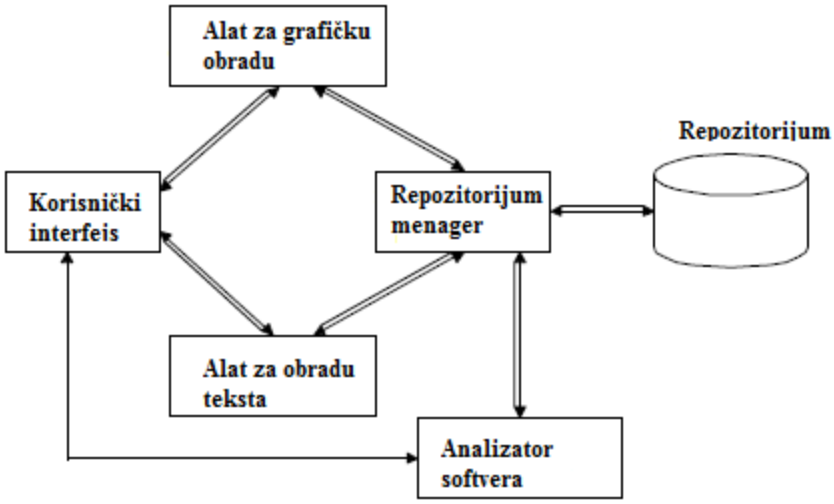
Алати за графичко представљање - Омогућавају развој различитих типова дијаграма на основу раније дефинисаних стандарда

Софтвер за дефинисање текста - Омогућава дефинисање назива, садржаја и детаља ставки у репозиторијуму

Анализатор софтвера - Анализира улазе у дијаграм и репозиторијум и утврђује њихову лексичку комплексност и проверава компатибилност са осталим објектима

Софтверски интерфејс према репозиторијуму -Обезбеђује везу између дијаграма и репозиторијума

Кориснички интерфејс - Обезбеђује интерактивну и off-line обраду



слика 7. CASE архитектура

Идеални CASE треба да омогући потпуну аутоматизацију комплетног животног циклуса пројекта информационог система, обухватајући почетну иницијативу, ниво анализе, рад на одржавању софтверског производа све до његовог повлачења. Такав CASE постаје жижа за све послове софтверског инжењерства, па се рад на развоју софтвера концентрише на логички аспект пројектовања. Управо због тога идеални CASE треба да омогући:

- израду архитектуре процеса организације,
- планирање и праћење пројекта,
- групни рад на развоју софтвера,
- апликативно и ручно дефинисање процедура,
- нормализацију података,
- генерисање шеме базе података,
- генерисање кода у кориснички одабраном програмском језику,
- аутоматско тестирање генерисаног кода према спецификацији апликације и експертну процену савршености производа, као и начин корекције.

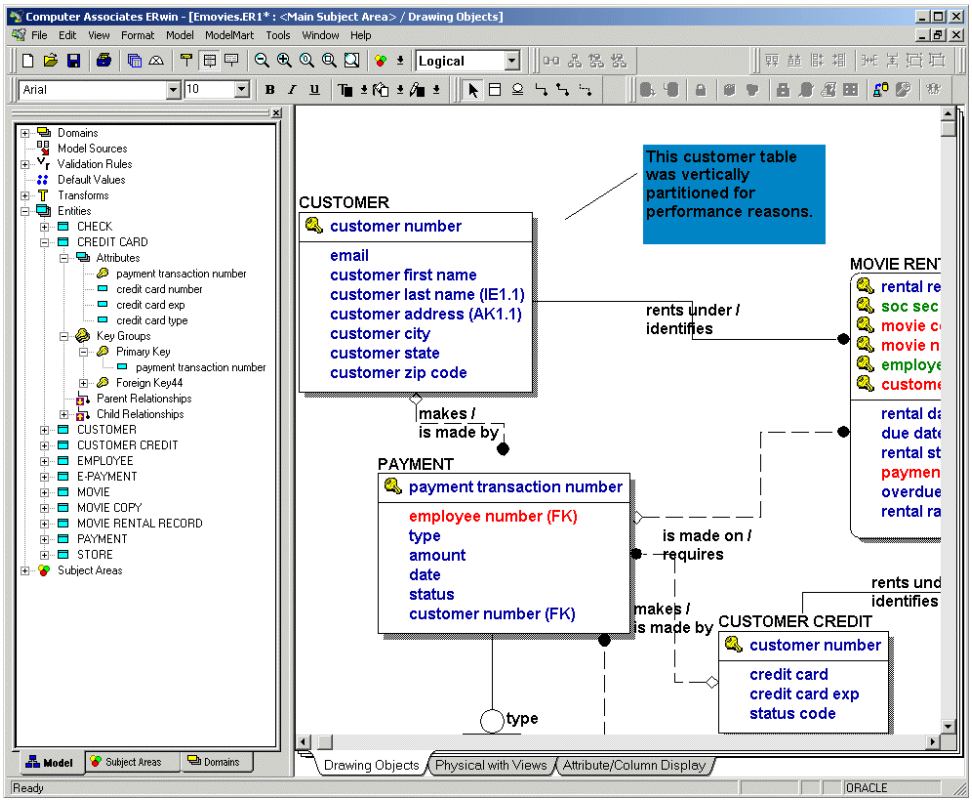
Идеални CASE треба већ у репозиторијуму да препозна компоненте које су за поновну анализу, дизајн и кодирање.

По правилу 40-20-40, које важи за развој програмског система, 40% пројектног времена отпада на анализу и моделовање, 20% је одвојено за програмирање, а преосталих 40% за тестирање. Тренд добављача је да се елиминише кодирање, што чини само једну петину времена потребну за развој софтверског производа. Садашњи тренутак захтева постојање CASE алата који ће покрити процес тестирања, а самим тим и скратити време израде софтверског производа. Будући CASE биће у могућности да у остатку 40% пројектног времена идентификује делове апликације који могу бити поново искоришћени. Из наведеног се може претпоставити и пут развоја CASE алата, а то је ефикасније покривање времена за анализу и дизајн, као и времена потребног за тестирање.

3.0. АНАЛИЗА ПОЈЕДИНИХ CASE АЛАТА

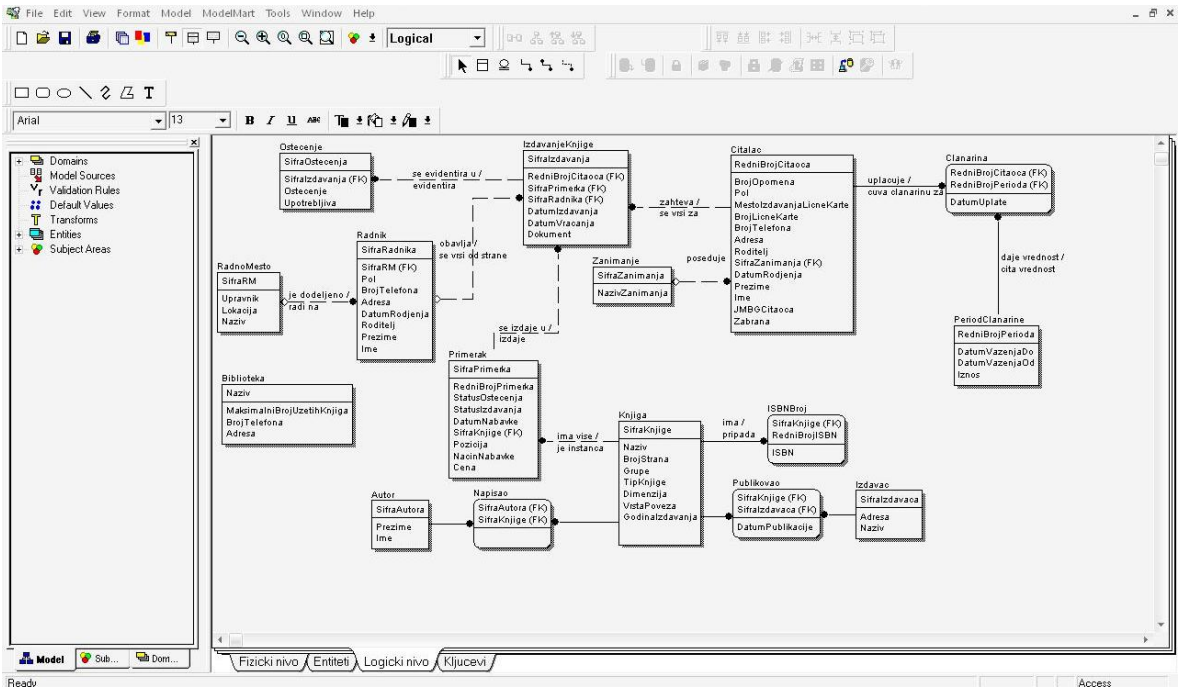
3.1 ERwin

ERwin алат, односно пуним називом AllFusion ERwin Data Modeler развило је предузеће Computer Associates International. Овај CASE алат служи за креирање и одржавање логичког и физичког модела података и генерисање њима припадајуће физичке базе података. Интерфејс је оријентисан кориснику, међутим рад са ERwin захтева висок ниво познавања поступака моделирања података.



слика8. кориснички интерфејс алата Erwin-у

У ERwin алату могуће је израђивати логички, физички или и логички и физички модел истовремено. Логички модел садржи ентитете, атрибуте који описују ентитете, везе међу ентитетима те примарне и секундарне кључеве. Физички модел садржи информације о физичкој бази подтака, као што су типови атрибута у релацијским шемама, ограничења, индекси као везе на друге релацијске шеме, денормализиране таблице и слично. Логичко физички модел садржи елементе и једног и другог модела.



слика 9. Логички модел података у Erwin-у

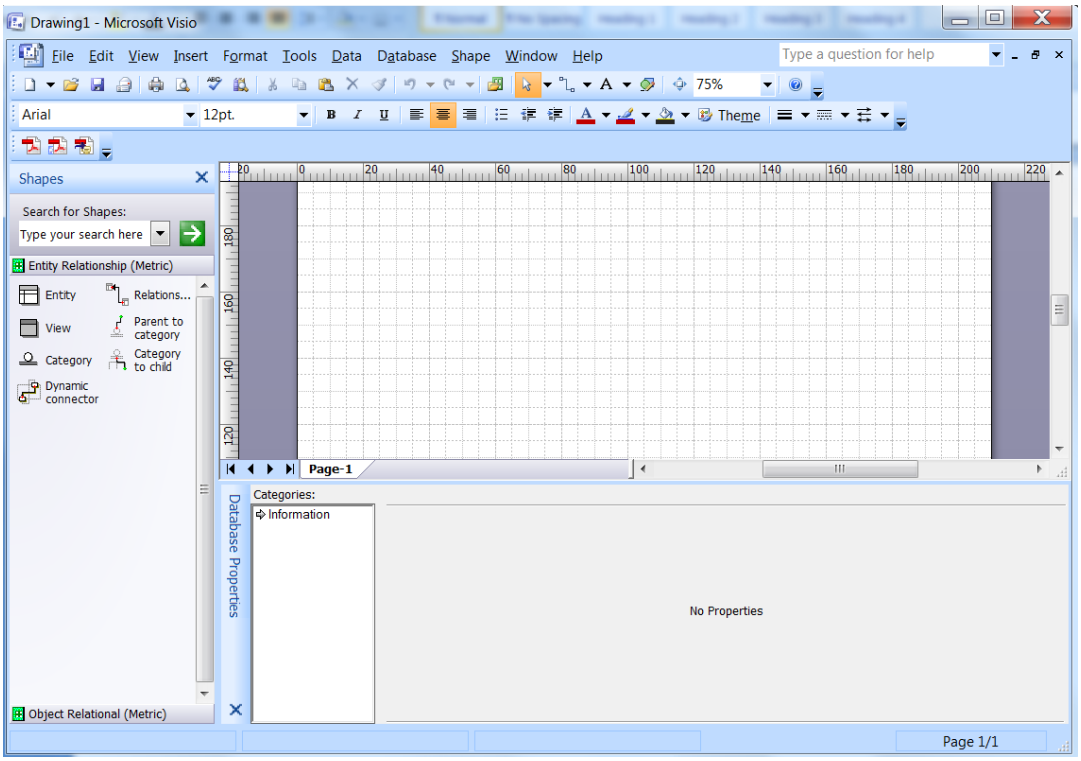
Свака датотека садржи базу у којој су похрањени попис и опис сваког ентитета. За сваки ентитет постоји скуп података о атрибутима који га описују (укључујући и примарни и секундарни/е кључеве), о везама према другим ентитетима, о уграђеним процедурама и окидачима за те процедуре и сл.

На темељу физичког модела података аутоматски је могуће генерисати и физичку базу података у коју се након тога уписују пословни подаци. CASE алат ERwin, подржава и обратну процедуру, а то је генерирање физичког, односно логичко/физичког модела података аутоматски из физичке базе података. Тај поступак назива се реверзно инжењерство.

Инверзно инжењерство је процес који обухвата информација из базе података о релацијским шемама, њиховим атрибутима, структури односно везама којима су релацијске шеме повезане. Након преноса информација у Erwin над насталим моделом података могуће је радити пренос које ће одговарати променама насталим у пословним подацима или њиховој структури и поновно сихронизовати физичку базу података са новим физичким или логичко/физичким моделом података. Цео процес могуће је и документовати алатком за извештај у ERwin -у.

3.2 Microsoft Visio

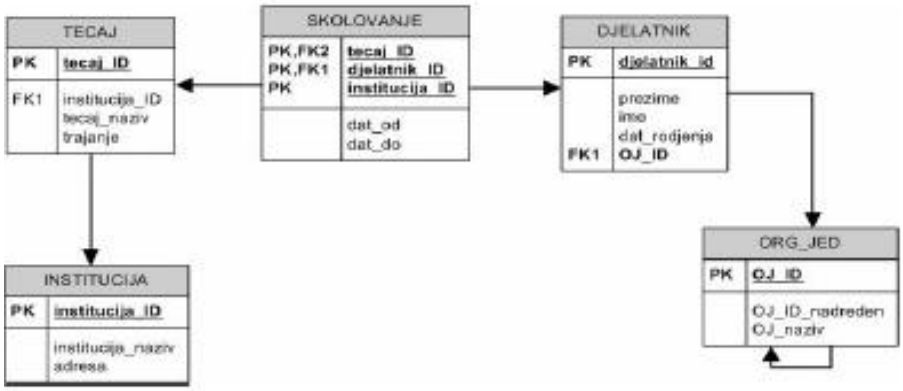
Visio је CASE алат, у склопу MS office, који у својој типичној инсталацији пружа могућности израде различитих модела. Сви дијаграми припадају једној од 16 категорија, или типова модела.Омогућава једноставно коришћење симбола за цртање и примери различитих модела омогућавају коришћење алата и особама које поседују мању количину знања о пројектовањз информационих система. На слици 10 . приказан је изглед почетног екрана за израду модела.



слика 10. Кориснички интерфејс алата Visio

Израда модела података могућа је у категорији за издраду дијаграма под називом Database цртањем Датабасе Database Model Diagram -а. Дијаграм садржи стандардне опције за израду ентитета, атрибута ентитета, кључних атрибута ентитета и различитих врста веза.

На слици 11. приказан је пример **Entity Relationship Model (ERA)** модела израђеног у Visio алату.

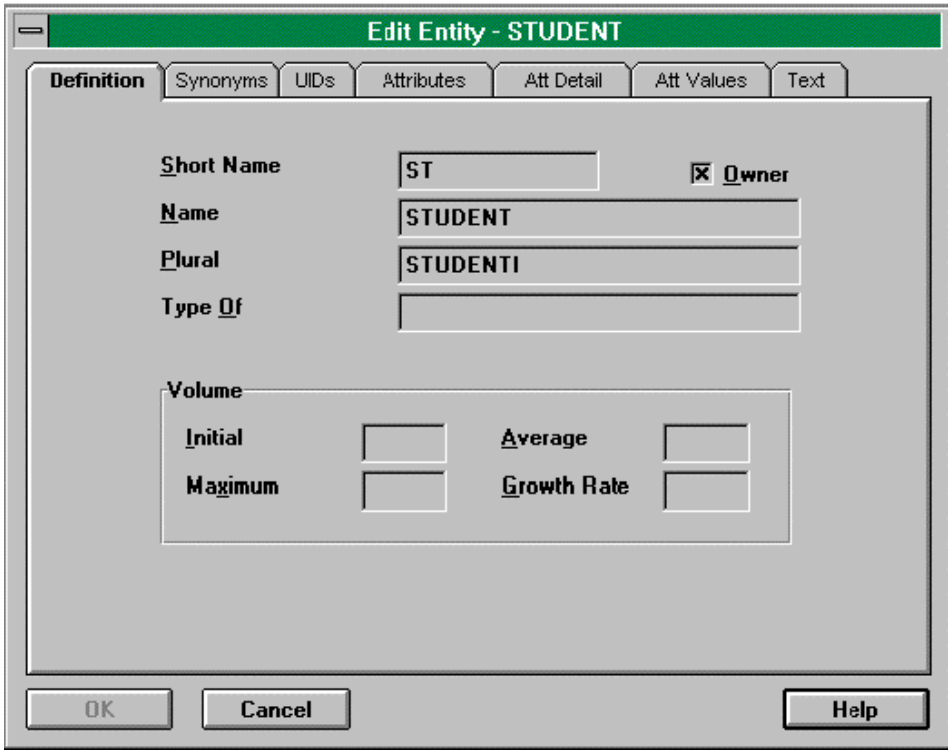


слика 11. ERA модел у Visio

Visio не пружа могућност израде репозиторија различитих модела, што значи да нема нити хоризонталне нити вертикалне повезаности модела. Пренос модела у друге програме омогућен је само у сврху презентације (word, excel, powerpoint) што значи да помоћу овог CASE алата није могуће нити аутоматско генерисање базе података нити обрнути поступак аутоматског генералисања модела података из базе података. Дакле није могуће издрадити модел у наведеном алату који би се користио за примену логичког модела података као средства за контролу произвођача софтвера како.

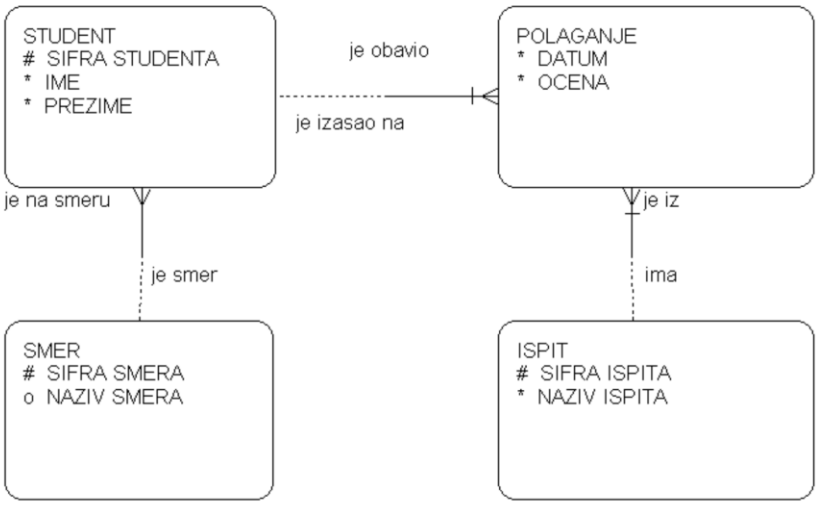
3.3. Entity Relationship Diagrammer (ERD)

Део *Oracle* пакета *Designer*/2000, који служи за моделирање података, подржава следеће концепте: атрибут, ентитет, специјализацију и генерализацију (подтипови и надтипови), везе. При креирању ентитета *ERD* захтева само његово име, а сви остали детаљи везани за ентитет, као и дефинисање његових атрибута, могу се касније обавити кроз опцију едитовања ентитета. По жељи се атрибути могу приказивати у оквиру ентитета на дијаграму и тада постоји конвенција да се: испред атрибута који улазе у састав примарног кључа ставља знак "#", испред атрибута који су обавезни ознака "*" и испред опционалних атрибута ознака "o", што доприноси прегледности дијаграма.



слика 12. EDR – екранска форма за дефинисање ентитета

При моделирању веза међу ентитетима задају се њихова опционалност (обавезност везе се представља пуном линијом, а необавезност везе испрекиданом линијом (са сваке стране везе понаособ) и њихова кардиналност (веза са горњом границом кардиналности 1 се представља једном тачком додира линије, која представља везу и одговарајућег ентитета, док се веза са горњом границом кардиналности **M** представља разгранатом линијом са три додирне тачке). Посебне врсте веза које су подржане овим CASE алатом су: преносиве и непреносиве везе (transferable i non-transferable), а односе се на могућност превезивања једне појаве ентитета са другом појавом референтног ентитета; спуштени кључеви, када се кроз овакву везу у кључ зависног ентитета укључује комплетан кључ ентитета од кога егзистенцијално зависи; рекурзивне везе, које моделирају везу једне појаве ентитета са другом појавом истог ентитета; ексклузивне везе, које моделирају ситуацију у којој може у једном тренутку постојати само једна од две или више веза које полазе од једног ентитета.



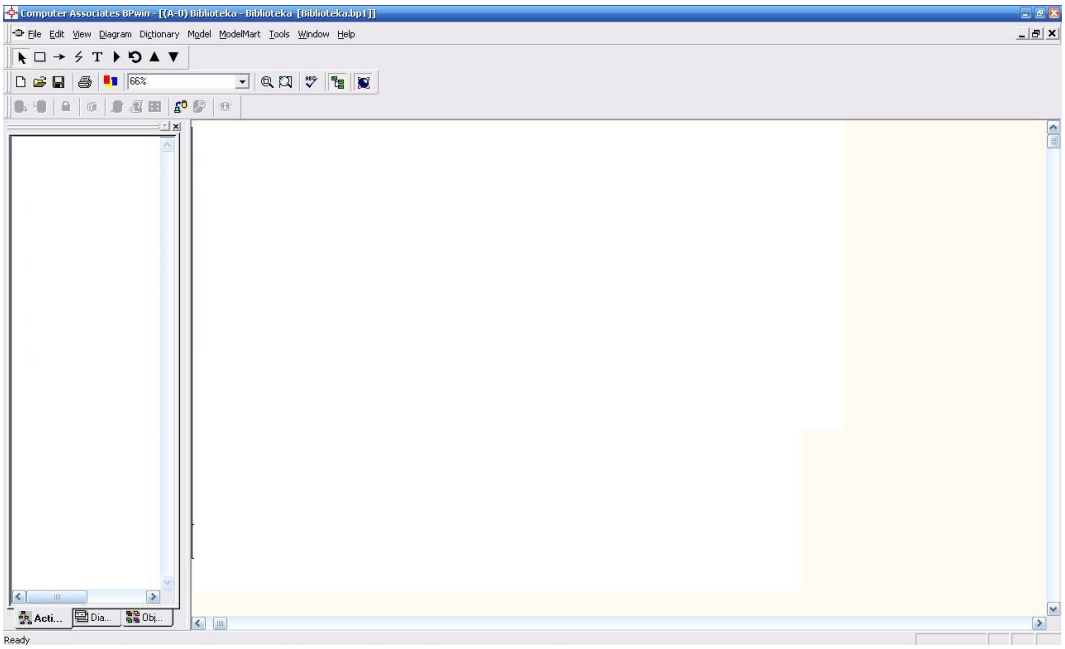
слика 13. ERD – пример модела објекти и везе у графичком едитору

Дефинисање домена за *ERD* се врши у *Repository Object Navigator*-у, алату за директан приступ речнику података. Домени дефинисани на тај начин су доступни и осталим модулима *ORACLE* -овог интегрисаног CASE-а, а у ERD-у се користе за дефинисање атрибута. Сваки објекат у ERD-у има карактеристику *CREATE* коју пројектант мења у *True* у тренутку када је завршио дизајнирање тог објеката. На тај начин се разликују објекти који улазе у генерисање излаза од објеката који не улазе, односно од оних који су још у фази дизајнирања. Приликом активирања анализатора грешака, објекти чији је *CREATE FLAG* једнак *False* се не разматрају.

3.4. BPwin (Bussines Process Windows)

BPwin алат за моделирање и анализу сложених пословних процеса је заснован на концептима стандарда IDEF0 . Због лакше идентификације процеса и њихове оптимизације, као и одбацивања сувишних процеса, настала је методологија моделирања пословних процеса. Моделирањем процеса се добија уређена структура са јасно дефинисаним правилима који се процеси одвијају и на који начин.
BPwin омогућава ригорозно тестирање конзистентности дијаграма. Динамички објекти који воде кроз модел спречавају грешке које најчешће настају при креирању модела. BPwin подржава ABC (Activity Based Costing) систем и има интерфејс ка ABC алатима намењеним компанијама које менаџмент стратегију базирају на активностима. Креирање документације из BPwin -ових модела процеса је изведено на исти начин као и у свим алатима фамилије LogicWorks-а. Постоји потпуна слобода у дефинисању сета података из модела активности, као и у дефинисању изгледа документације. BPwin

омогућава рад у графичком окружењу, уз пуну подршку рада мишем, тако да је едитовање модела података изузетно лако; дијаграми који се добијају дају потпуни приказ модела. Декомпозиција процеса се врши директно на дијаграму, а кретање по нивоима декомпозиције подразумева пребацивање са дијаграма на дијаграм. При раду на једном моделу отворени су сви дијаграми из тог модела.

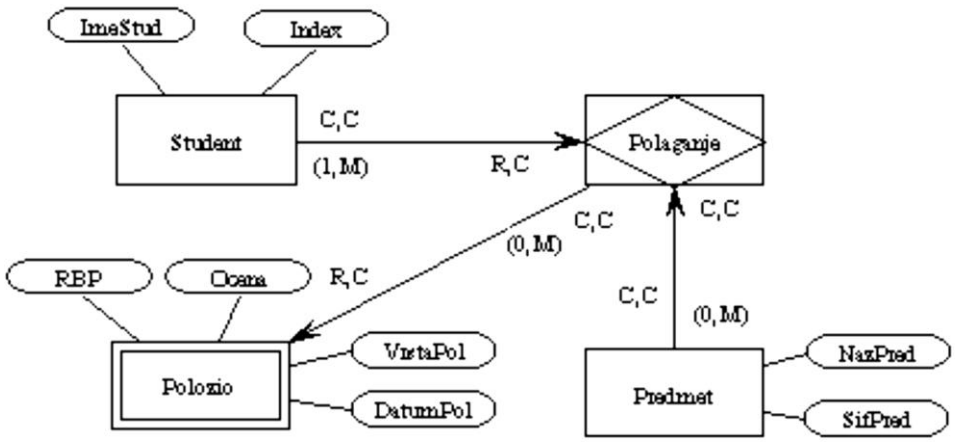


слика 14. Изглед корисничког интерфејса алата BPwin

3.5. Artist

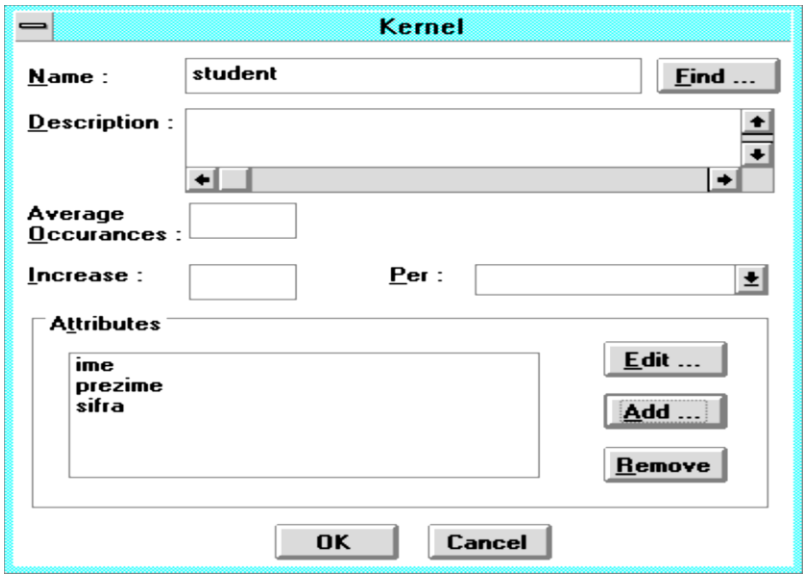
Artist CASE алат је развијан да би се покрио целокупан развој информационог система. Алат омогућава аутоматизацију прве фазе у развоју информационих система, односно планирање развоја информационих система, које се заснива на *BSP* (*Business System Planning*) методи.

Поред тога, поседује и модел који подржава моделовање процеса (*SSA*). Модул који омогућава моделовање података подржава семантички веома богат модел, тако да се може моделирати неки скуп података на веома ефектан и брз начин. Артист подржава следеће концепте: јак ентитет, слаб ентитет, агрегацију, специјализацију и генерализацију (подтипове и надтипове), везу, идентификујућу везу. Део концепата приказан је а примеру (слика 15).



слика 15. Artist – пример модела "објекти и везе" у графичком едитору

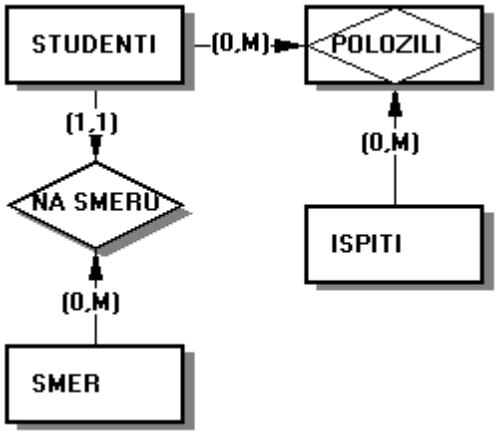
Изглед атрибута и опис ентитета кроз форму дати су, такође, на слици 7. Нема команди за додавање, брисање и исправку атрибута на самој форми за опис ентитета, већ се то реализује на форми која се активира командом *Attributes*, након чега се појављује форма за унос (слика 16.)



слика 16. Artist CASE – екранска форма за унос података о ентитету

3.6. MagiCASE

MagiCASE je графички оријентисан алат за моделовање података. Заснован је на следећим концептима: атрибут, јак ентитет, слаб ентитет, агрегација, специјализација и генерализација (подтипови и надтипови), веза, идентификујућа веза. У самом алату може се мењати графичка презентација концепата, чиме је корисник донекле ослобођен крутих стега методологије, што представља велики подстрек у креативности корисника програма.

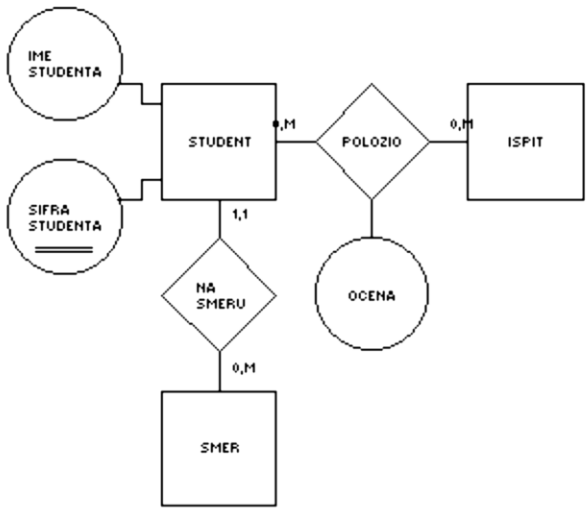


слика 17. MagiCASE –пример модела “објекти и везе” у граафичком едитору

Код Kod MagiCASE-a у речник података уведен је и појам домена, чиме је омогућено да се атрибути, чији су типови исти, могу дефинисати преко претходно креираних домена.

3.7. EasyCASE System Designer

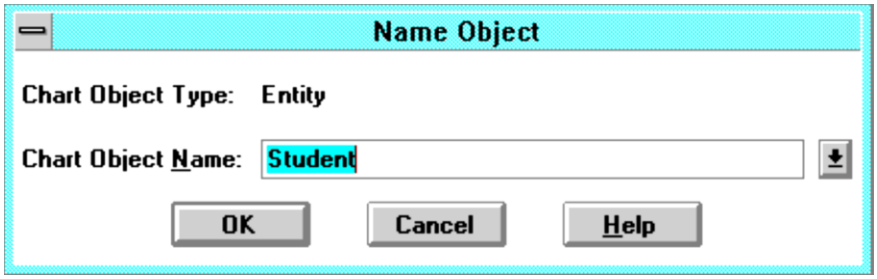
EasyCASE System Designer је графички оријентисан алат за развој система који ради само у *Microsoft Windows* окружењу. Помоћу скупа модула могу се моделирати процеси по методологијама: *Yourdan*/*DeMarco, Gane* & *Sarson, SSADM*, а, исто тако, и моделирање података по методологијама: *Chen, Martin Bachman, IDEF1X, Shlaer* & *Mellor*.



слика 18. EasyCASE – пример модела "објекти и везе" у графичком едитору

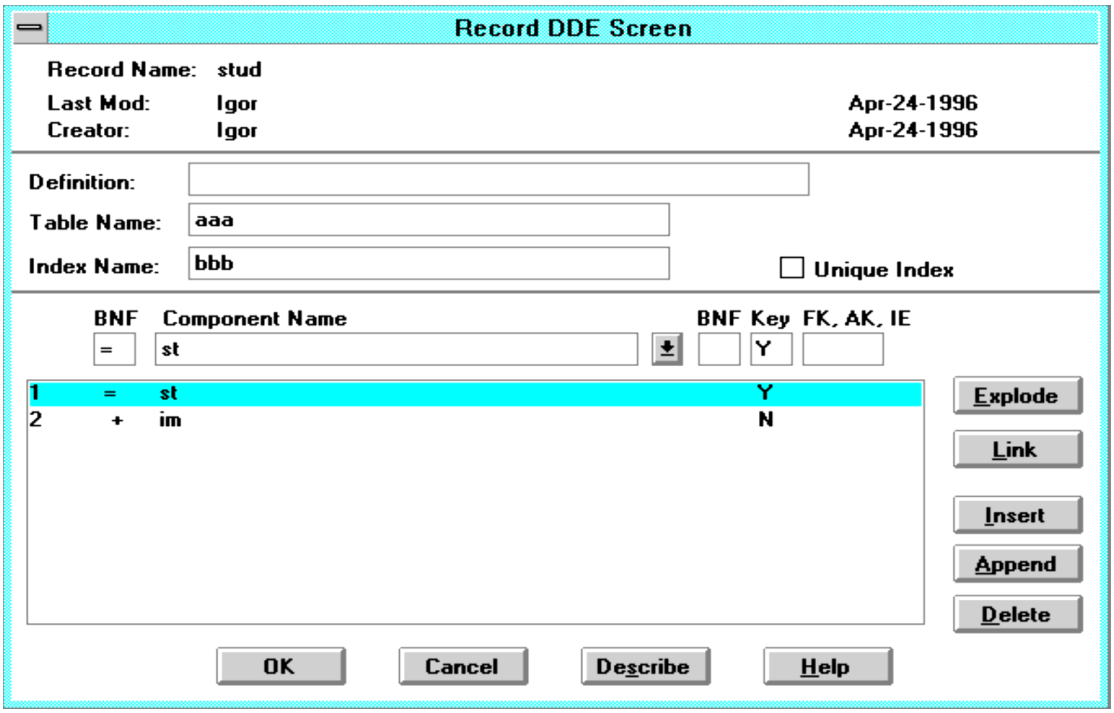
EasyCASE System Designer подржава следеће концепте: атрибути (за које се мора нагласити да ли је неки од њих обичан атрибут), примарни кључ, спољни кључ, вишевредносни или изведени атрибут, јак ентитет, слаб ентитет.

Део концепата дат је примером на слици 18.
Док је на слици 19. дата је форма за унос ентитета.



слика 19. EasyCASE – екранска формаза дефинисање ентитета

Атрибути се у подмоделу морају дефинисати одвојено као самосталан објекат, независно од ентитета, након чега се атрибут може повезати са ентитетом коме припада. Дефинисање спољног кључа као типа атрибута представља редундантни податак, имајући у виду да је појављивање спољних кључева дефинисано успостављеним везама.
Други начин креирања атрибута је преко дефинисања *child object*-а. Активирањем одређеног концепта и дефинисањем његовог *child object*-а, као као *record*-а , понуђено је да се дефинишу и његове компоненте, односно атрибути концепта.

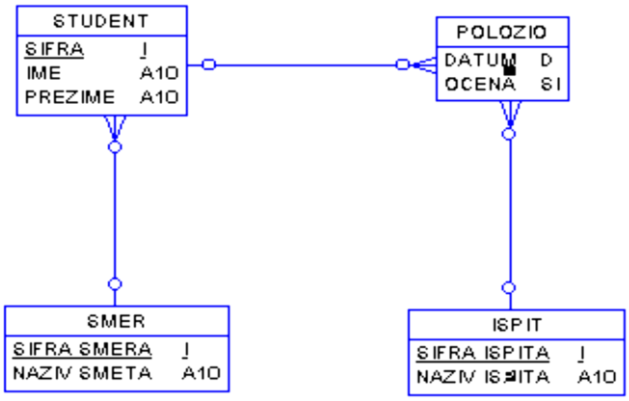


слика 20. EasyCASE –екранска форма за дефиниање атрибута

Подржава генерисање шеме базе података за следеће сервер базе: *ORACLE7*, *DB2*, *Rdb*, *Ingres*, *SQLServer*, *SQLBase*, *SYBASE*, *Progress*, као и системе за управљање базама података на персоналним рачунарима: *Clipper*, *FoxPro*, *dBASE*, *Paradox*

3.8. S-Designor

S-Designor је алат за моделирање података, графички оријентисан, који ради у окружењу *Microsoft Windows*. . Основни елементи на којима почива су: атрибути, ентитети, везе. Део концепата које подржава *S-Designor* дат је на слици 21.



слика 21. S-Designor – пример модела “објекти и везе” и графичком едитору

Иако привидно сиромашан (поседује само један концепт од објеката), могу се исказати све врсте концепата које се појављују у моделу објекти-везе. Приликом дефинисања ентитета и његових особина графичке презентације, рад са њим је идентичан, независно од тога који се концепт представља. Назив, атрибути и описи уносе се на идентичан начин код свих објеката, а разлике између њих постају очигледне тек када се определи за врсте веза којима се одређени објекат повезује са остатком подмодела.

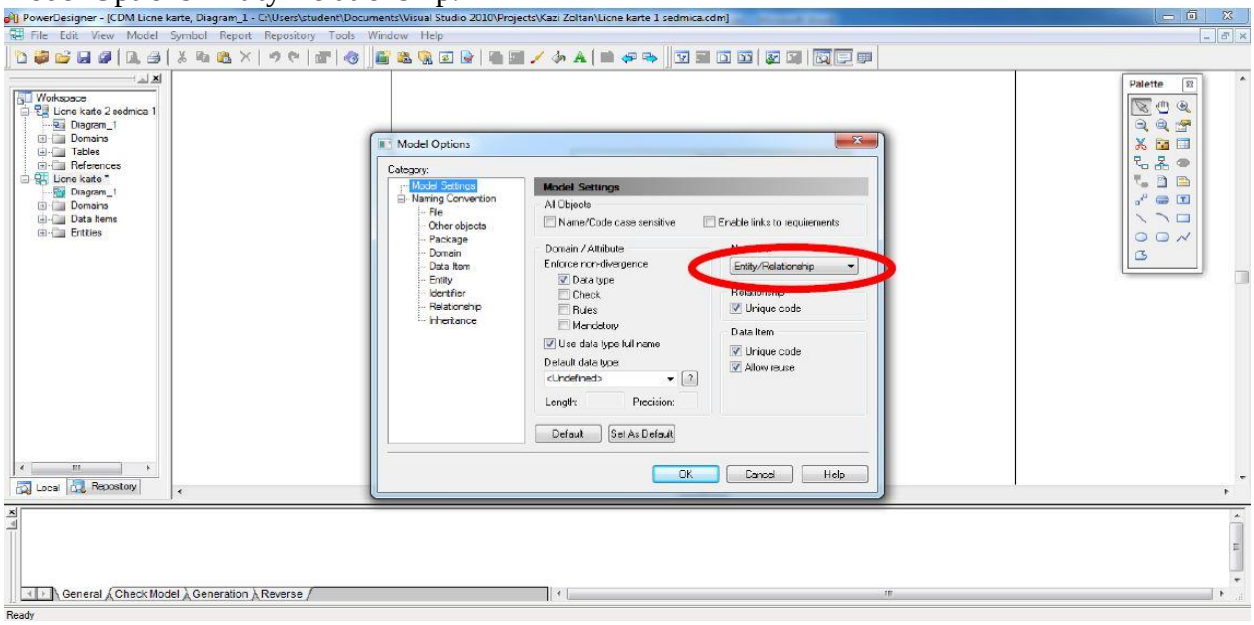
3.9. ADW (Application Development Workbench)

ADW је *KnowledgeWare*-ov *CASE* алат са интегрисаним деловима који омогућава аутоматизацију животног циклуса развоја информационог система. Први део *Planing* Workstation подржава дефинисање објеката, као и њихово повезивање преко асоцијативних матрица. Могу се дефинисати организациона структура и информационе потребе, циљеви, пословне функције, критични чиниоци успеха. Постоји могућност утврђивања приоритета пројеката на основу постављених критичних чинилаца. Све информације се смештају у централну енциклопедију којој могу да приступе сви делови алата.
Други део *Analisis Workstation*, који служи за моделовање процеса и података, садржи читав низ модула за цртање дијаграма: дијаграм саставних делова, дијаграм тока података, *ER* дијаграм, дијаграм типова информација, као и акциони дијаграм. Део *RAD Workstation* је алат за израду прототипова и омогућава испитивање почетног дизајна апликација за крајњег корисника.

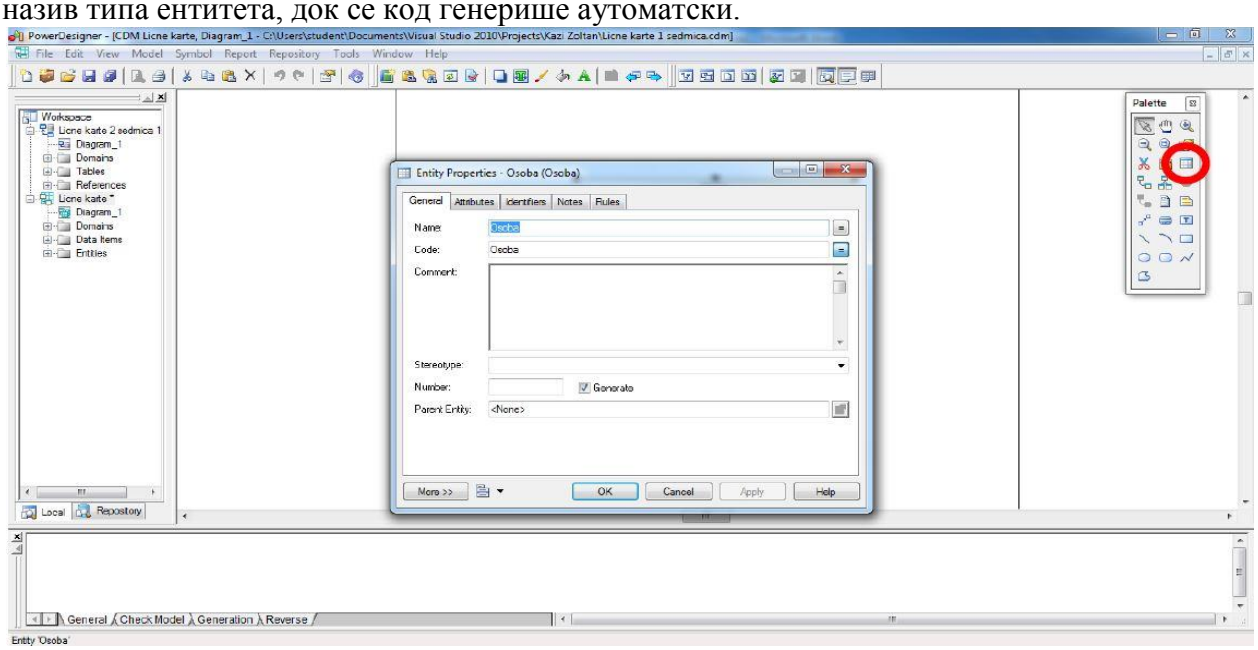
Делови *Design Workstation* и *Construction Workstation*, на основу претходних информација које су смештене у енциклопедији, могу изгенерисати генеричке клијент/сервер-апликације релационих и хијерархијских база података (*Oracle7, Sybase SQL Server, DB2, DB2\2, IMS, VSAM*) и релационе базе које подржавају *ANSI* стандард. Генератори на основу генеричке апликације израђују клијент/сервер-апликацију за циљни оперативни систем (*UNIX.Motif, OS/2 Presentation Manager, DOS/Windows3.x, AS/400 i MVS*).

4.0. ПОСТУПАК КРЕИРАЊА БАЗЕ ПОДАТАКА ПОМОЋУ SYBASE POWER DESIGNER – СТУДИЈСКИ ПРИМЕР

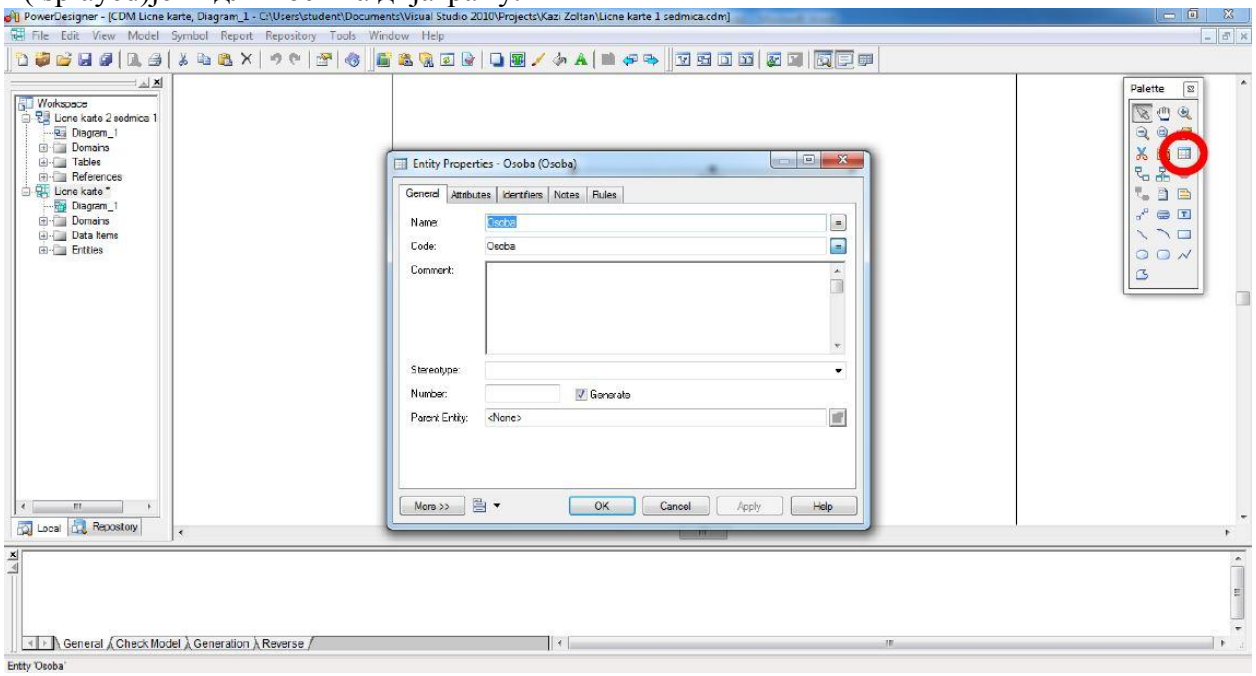
1. Покренути програм: Start-Sybase-Power Designer. Креирати нов концептуални модел података, опција је File-New Model-Conceptual Data Model. Подесити нотацију: Tools-Model Options-Entity Relationship.



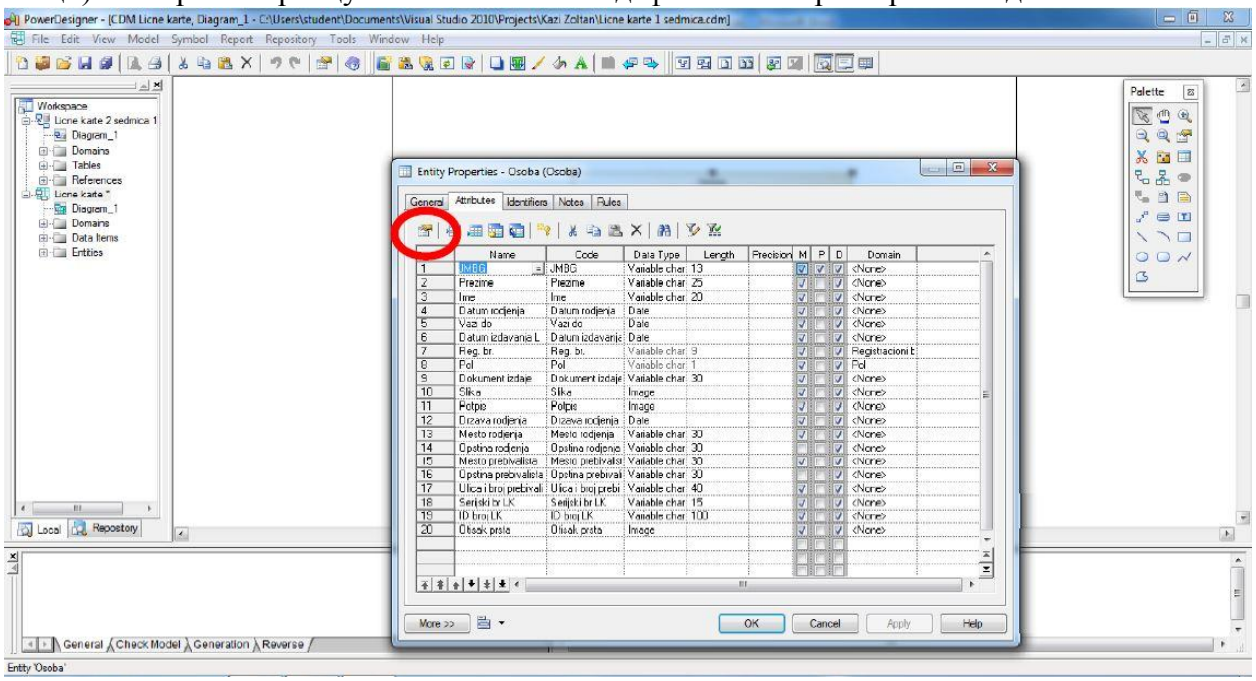
2. Креирање објекта/ентитета се врши помоћу алатке Entity, примењене са палете алата (укључивање алата је преко Tools-Customize Toolbars-Pallete=True). Уписати у Name поље назив типа ентитета, док се код генерише аутоматски.



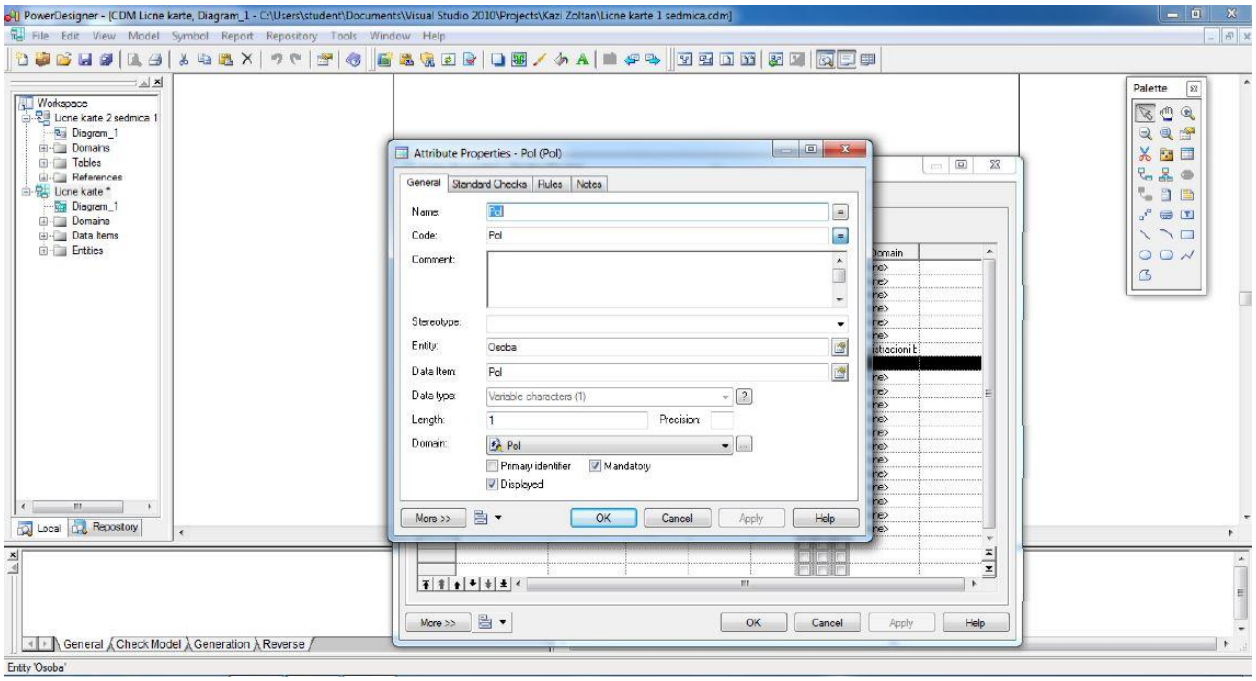
3. Дефинисати особине атрибута у ентитету, у оквиру картице Attributes. Уписује се назив, бира се тип података, дужина поља за унос и по потреби кориснички дефинисан домен. M(andatory)=True је обавезна вредност атрибута, P(rimary) је идентификациони атрибут, D(isplayed)је видљивост на дијаграму.



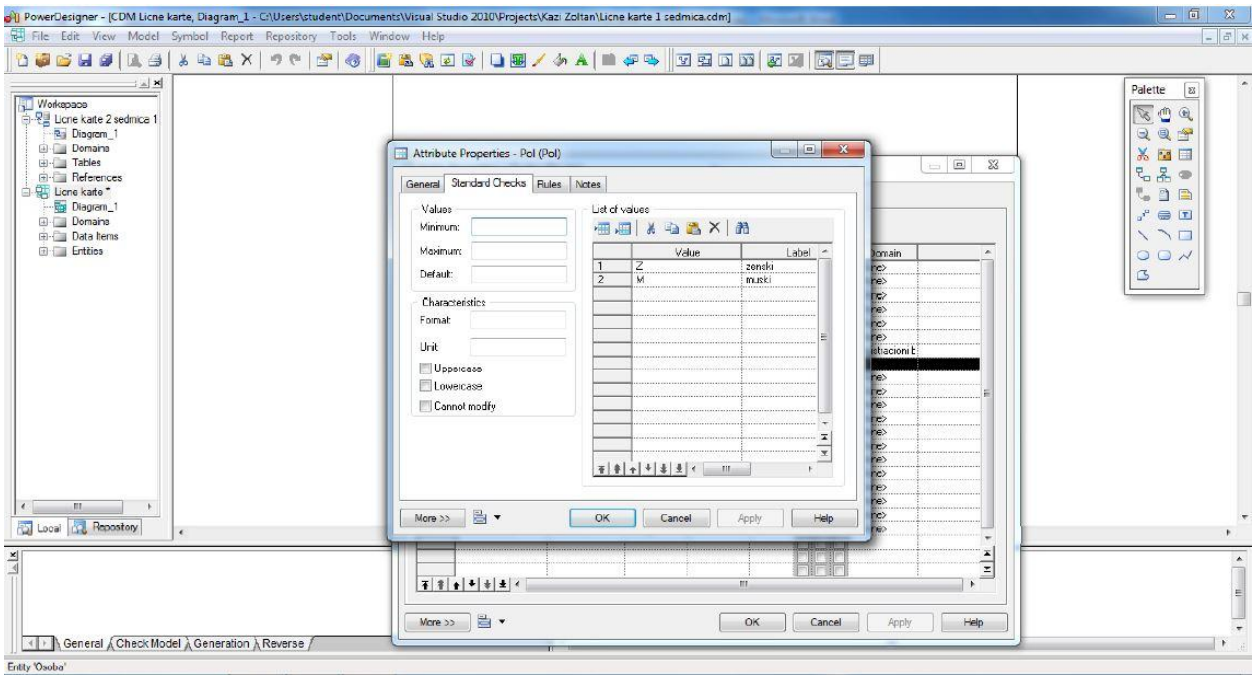
4.Креирање домена атрибута, опција Model-Domains (уписати назив домена) или Attribute Properties у оквиру ентитета, па у оба случаја Пропертиес (алат означен на претходној слици). Отворити картицу Standard Checks и дефинисати карактеристике домена.



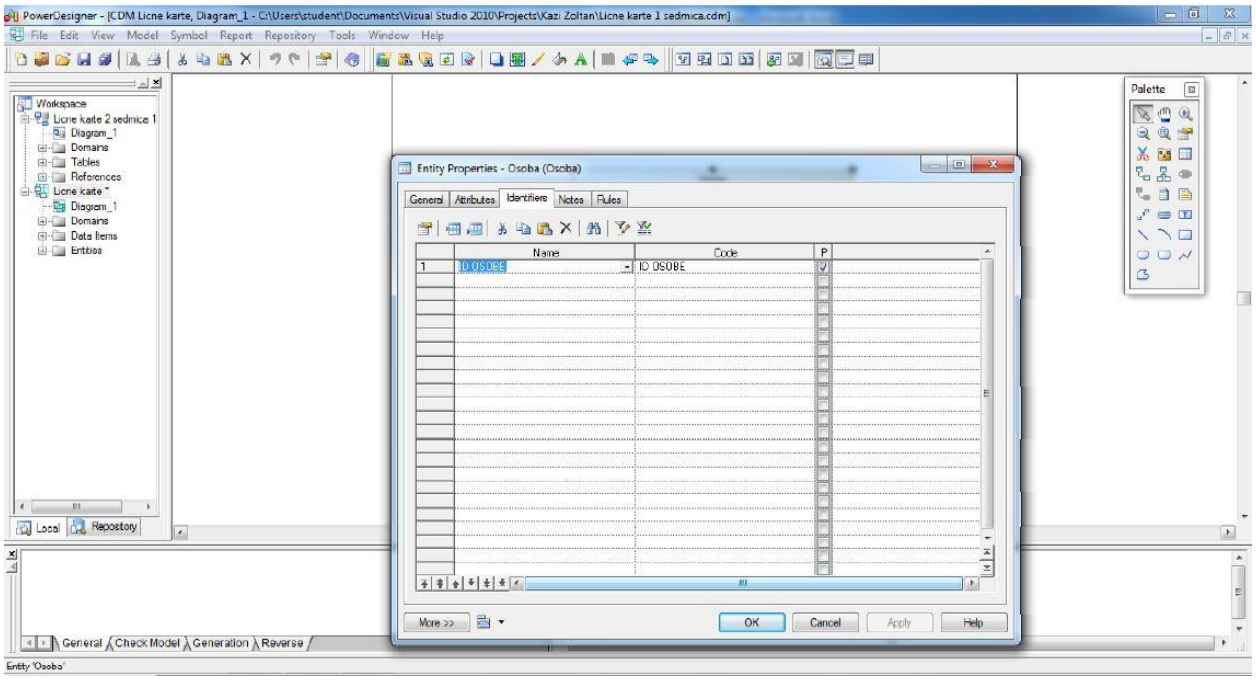
5. Креирање домена атрибута, наставак: Набројиви тип података уписати у Vlaue колони, са описом у Label; или Min. и/или Max. вредност; иницијалну вредност; формат уноса, корак; велима или мала слова или непроменљивост. У Rules картици се може формирати израз (са формулама и функцијама) за проверу вредности које ће корисници покушати да упишу у базу података.



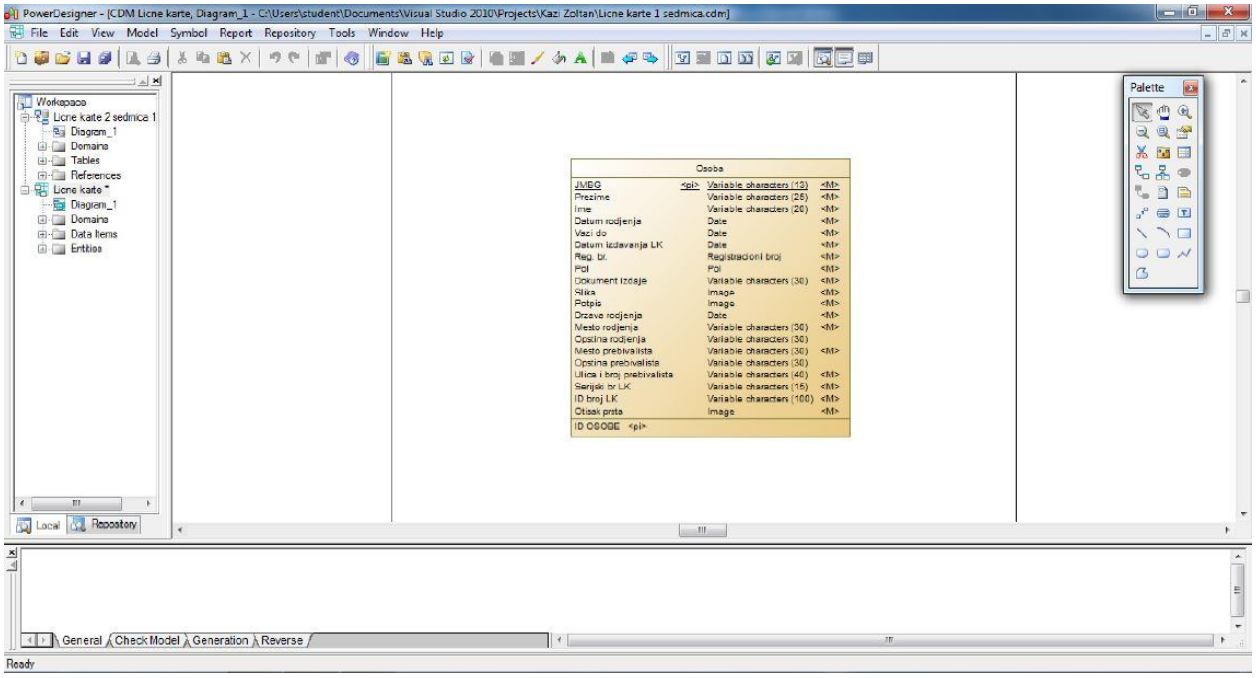
6. Одређивање имена идентификатора у чији састав може ући и више атрибута.



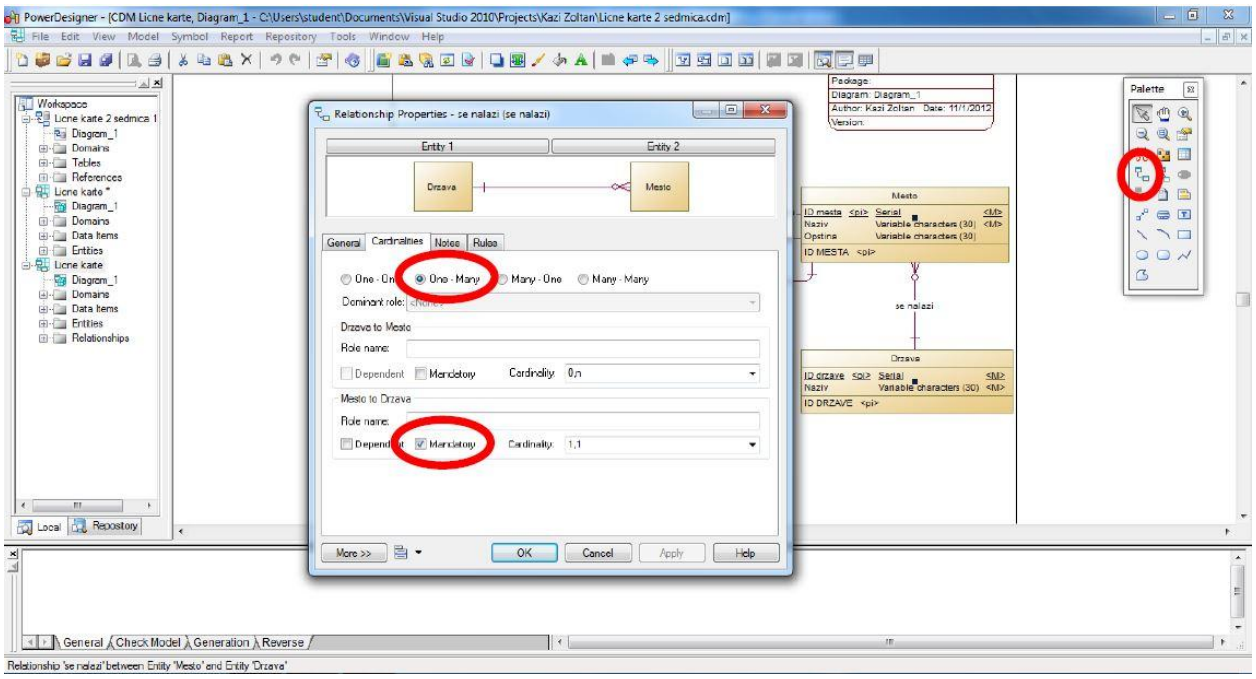
7. Формирани ентитет, на примеру: Особа. Поступак се понавља за потребни број ентитета који се жели формирати у моделу.



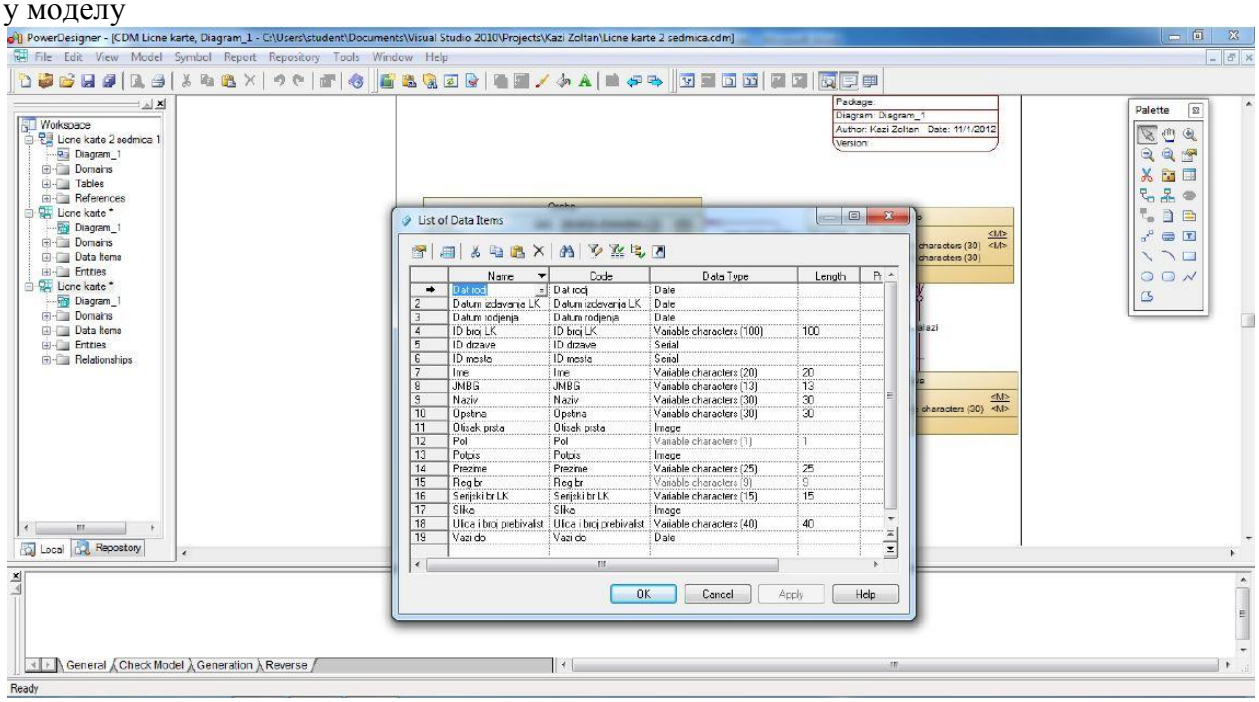
8. Успостављање повезника између ентитета, на примеру: Држава-Место. Примењује се алат Relationship са палете. У картици General се уписује назив релације, а у Cardinalities: Тип везе: 1-1, 1-M или M-M. Затим се одређује доња граница кардиналитета повезника тако што се особина Mandatory постави на True, уколико је доња граница=1.



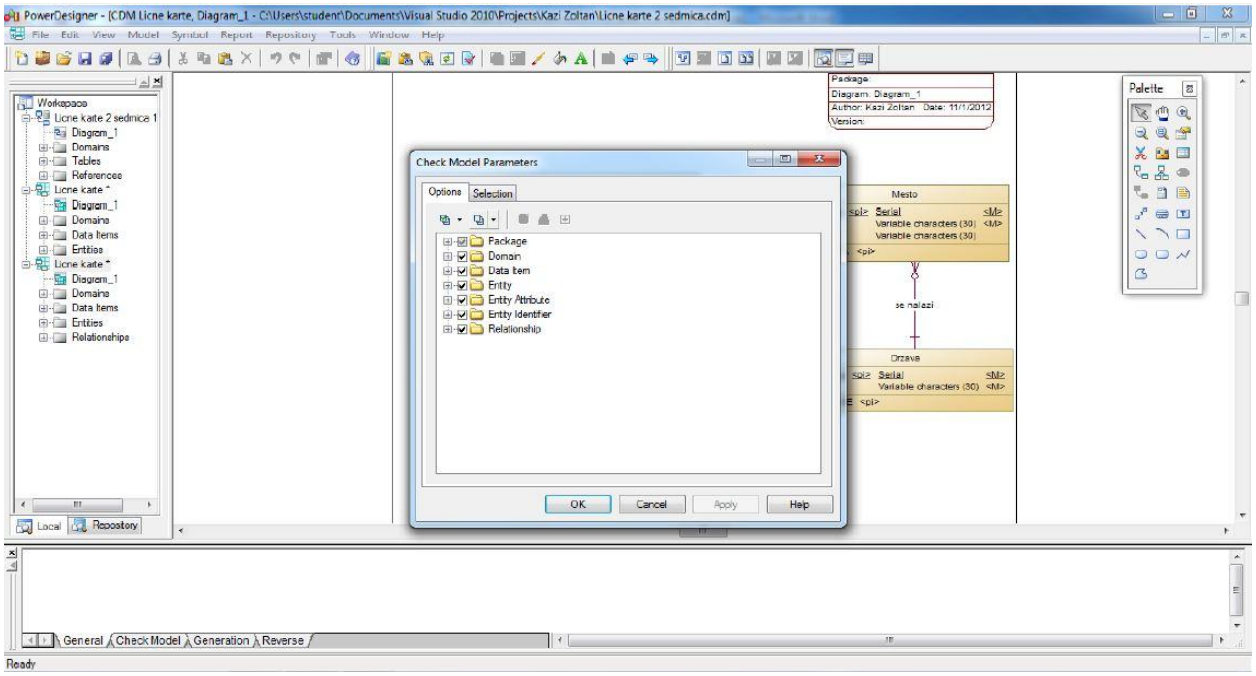
9. Након што су уведени нови ентитети и успостављене одговарајућа релације, потребно је обрисати из почетних ентитета атрибуте који припадају новоформираним ентитетима. Приказ креираног концептуалног модела података је дат на следећој слици:



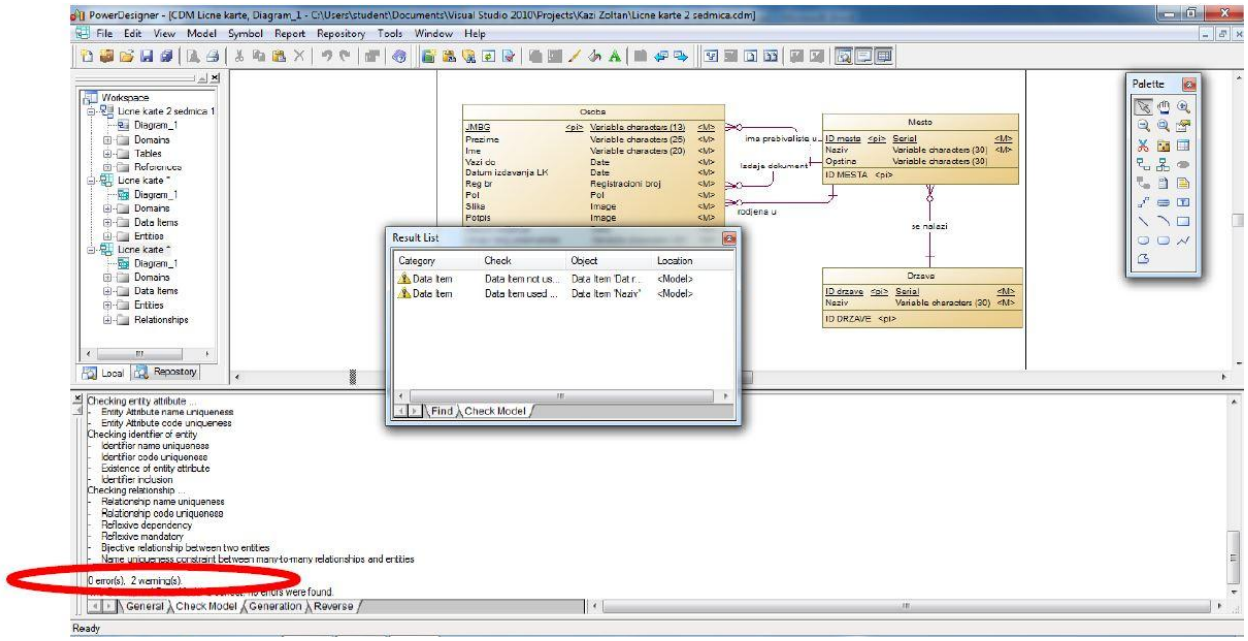
10. Брисање неупотребљених атрибута се врши приступом Речнику података (Model-Data Items, у филтеру укључити Used By особину) и обристи атрибуте које не користи ниједан ентитет. Такође је могуће централизовано одржавати и ажурирати особине свих атрибута у моделу



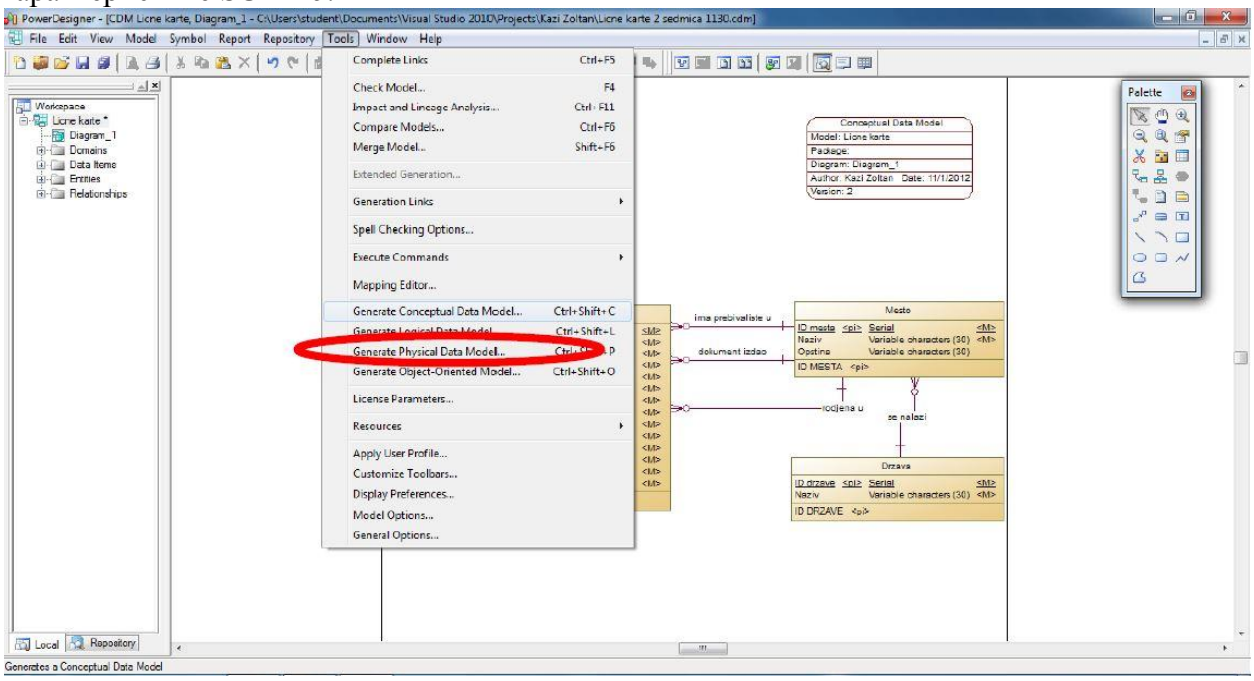
11. Провера исправности концептуалног модела података, који не сме да садржи грешке у моделовању (Tools – Check Model).



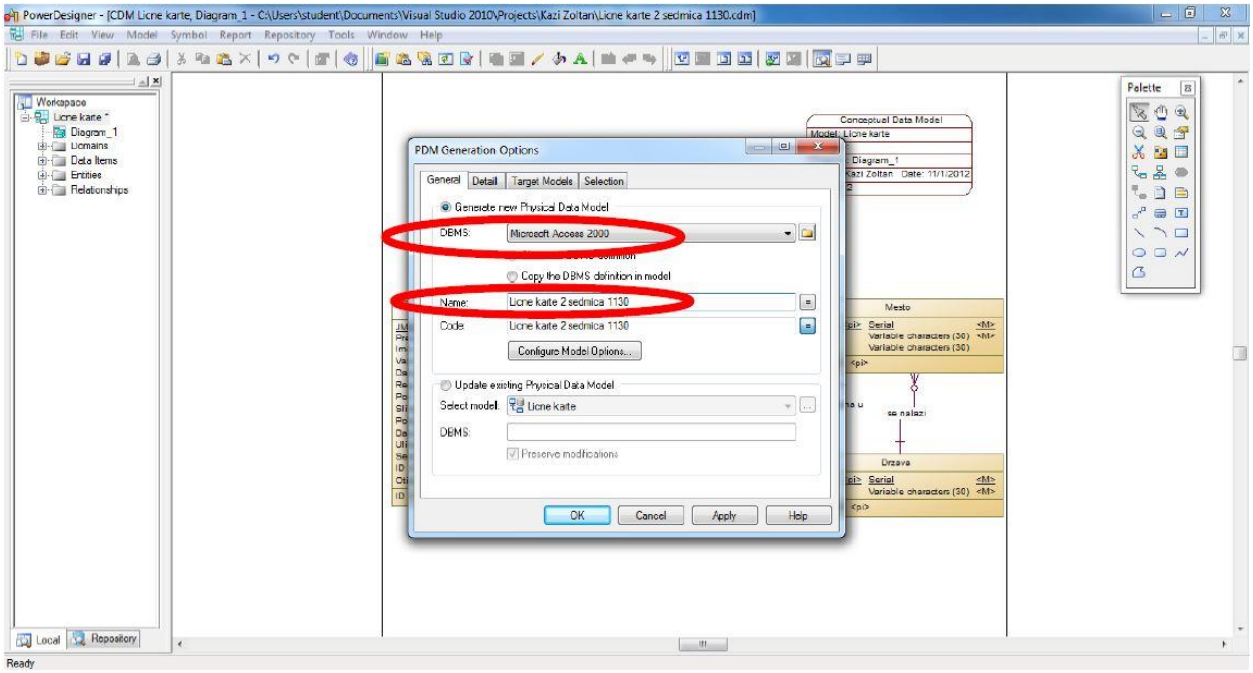
12. Поруке алата за проверу модела. Упозорења су дозвољена (проверити и извршити по потреби корекције).



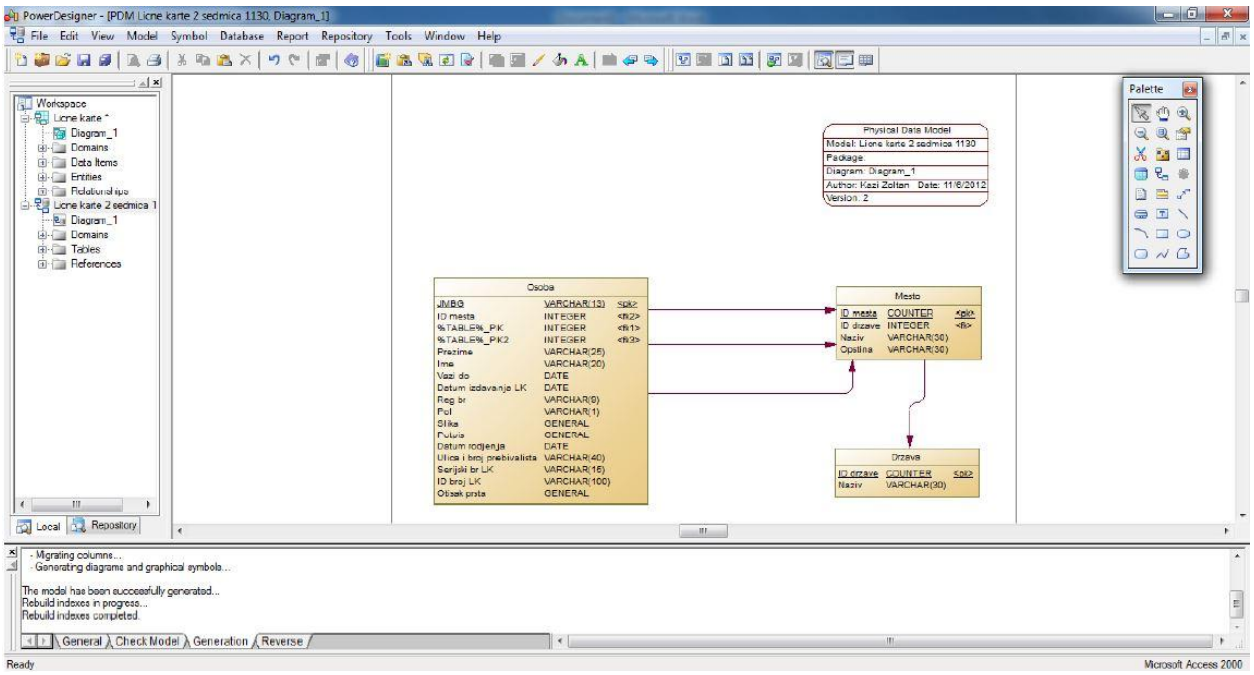
13. Креирање физичког модела података, изабрати опцију менија: Tools-Generate Physical Data Model.. Алтернативно се може прво урадити Релациони модел независтан од имплементације у конкретном SUBP-u, па тек тада Physical Data Model, који садржи карактеристике SUBP-u.



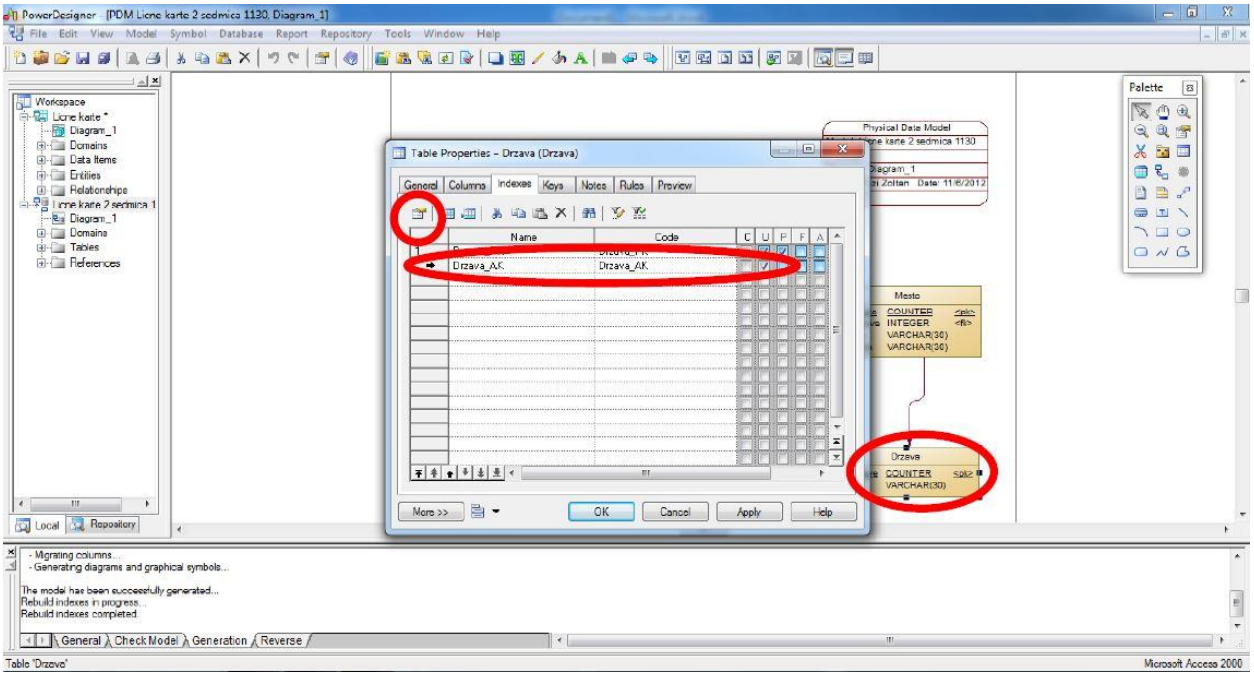
14. Креирање физичког модела података, корак два, проверити или уписати назив модела и изабрати одговарајући DBMS (SUBP), тј. софтвер за управљање базама података у којем се жели имплементирати модел. У картици Detail се одређују начини именовања објеката у физичком моделу и одређују правила за очување референцијалног интегритета.



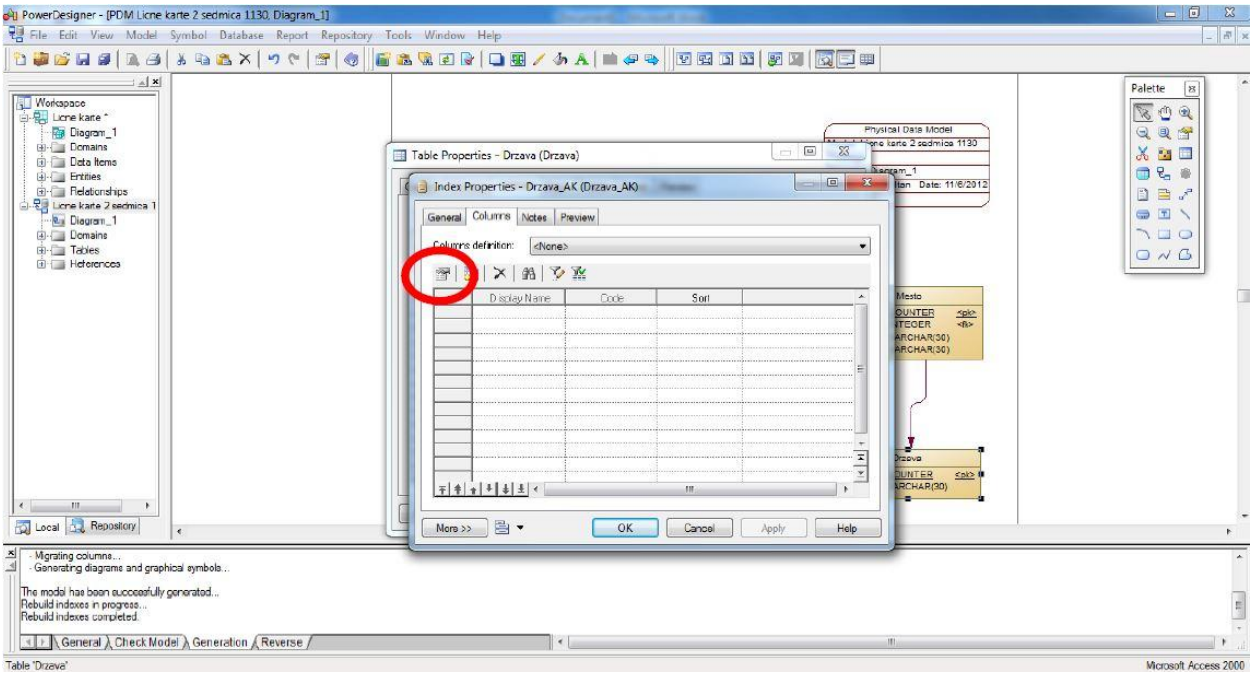
15. Креирани физички модела података приказан је на следећој слици:



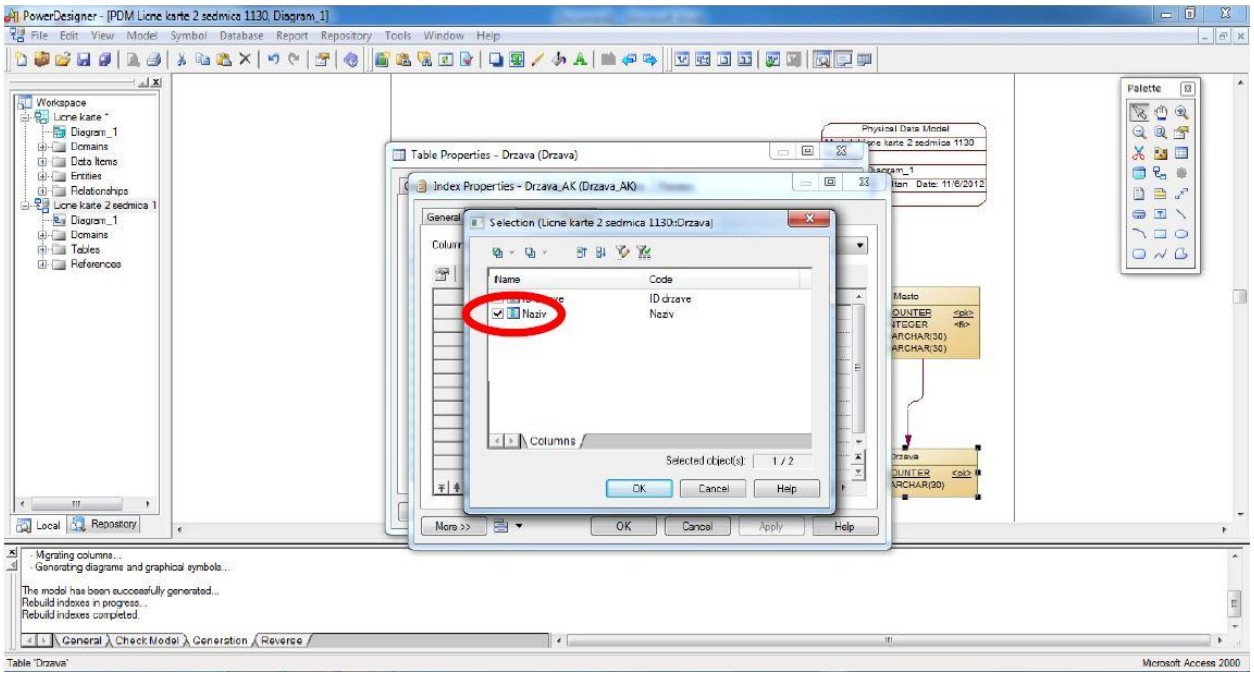
16. Дефинисање алтернативних кључева у случају да се жели спречити редунанса података (нпр. за табелу Држава), отворити прозор за дефинисање особина табеле, уписати назив алтернативног кључа, изабрати особину Unique=True, па се затим примењује Properties алатка.



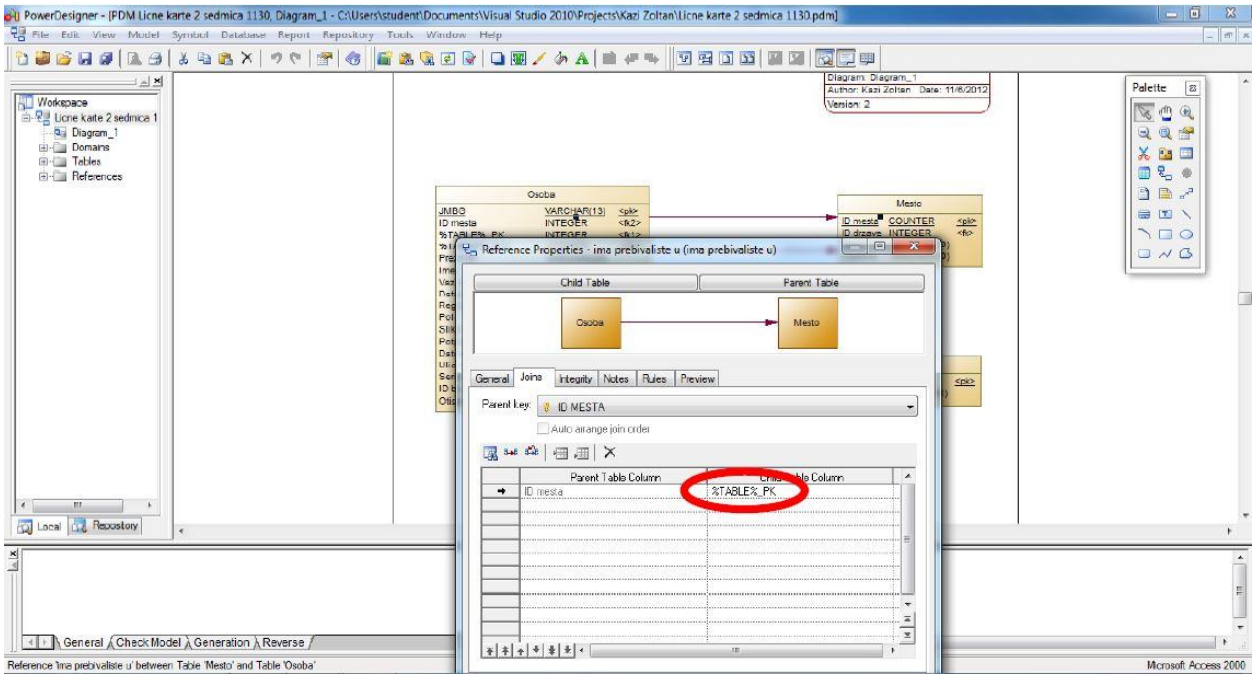
17. Додавање колона из табеле које ће чинити алтернативни кључ, опција Add Columns.



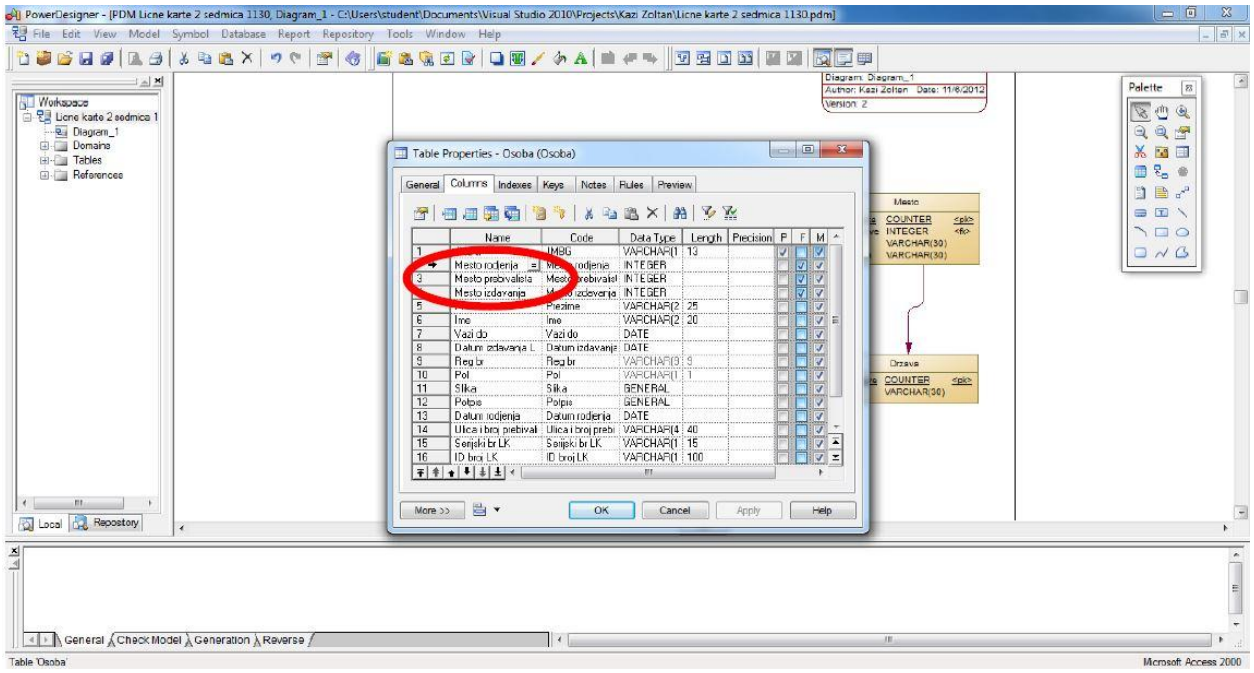
18. Избор колона из табеле за алтернативни кључ.



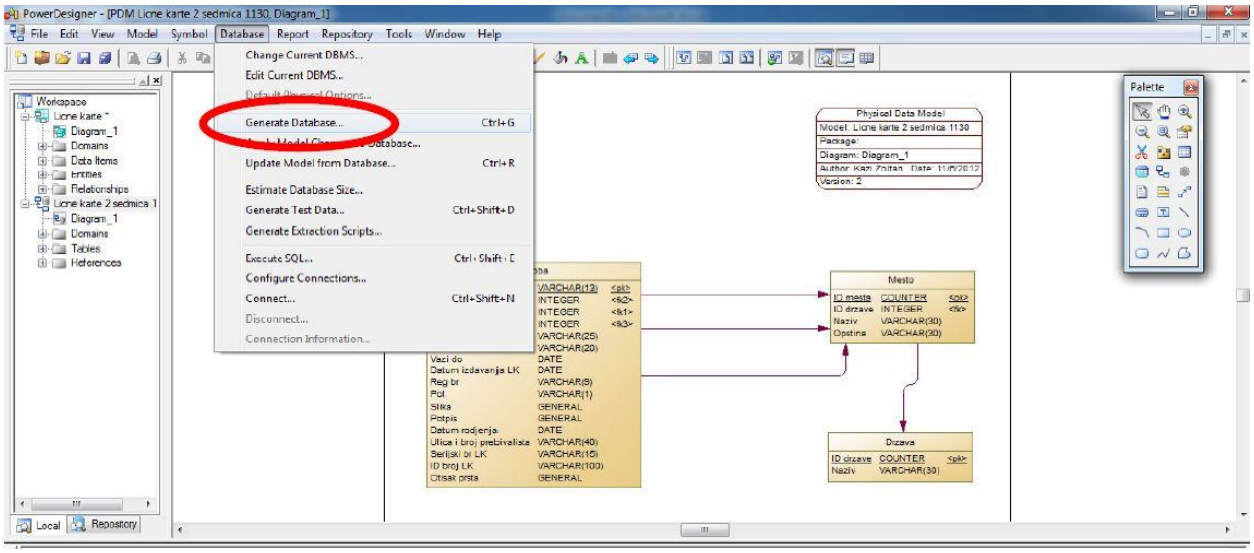
19. Преименовање колона уколико назив није одговарајући (нпр. код страних кључева). Отворити особине везе између табела и погледати природу односа табела у насловној линији:



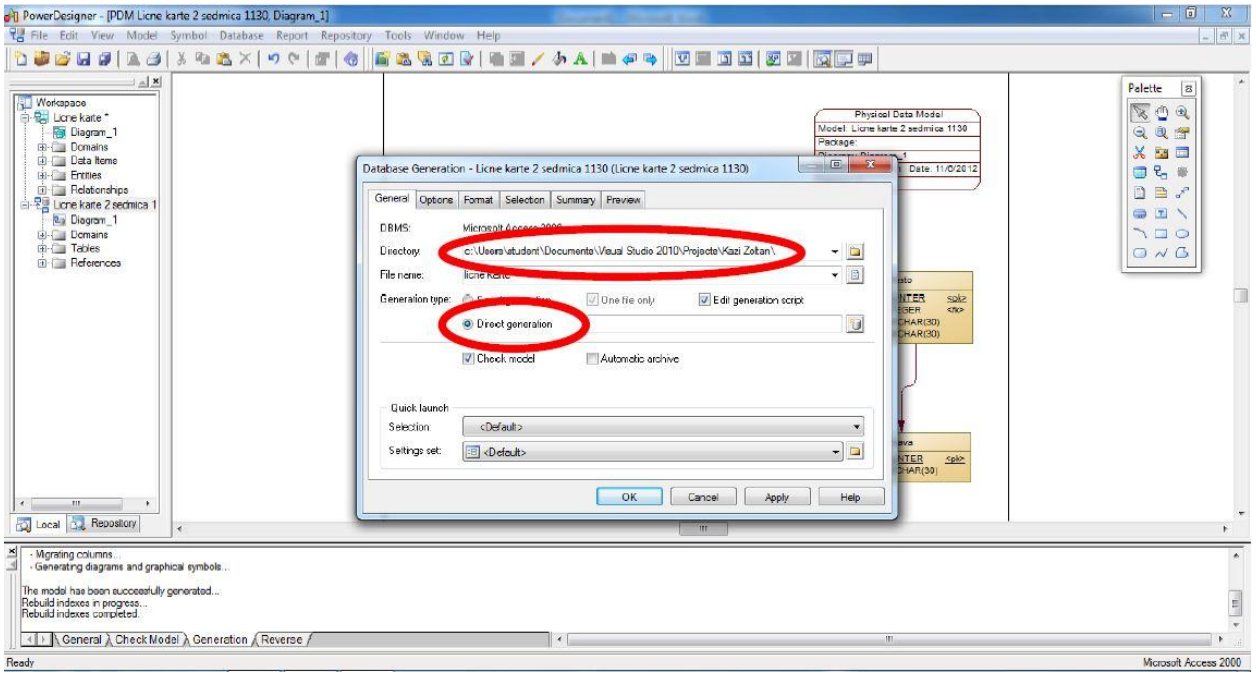
20. Промена назива колона на примеру табеле Особа - код страних кључева: Место рођења, Место пребивалишта и Место издавања личне карте.



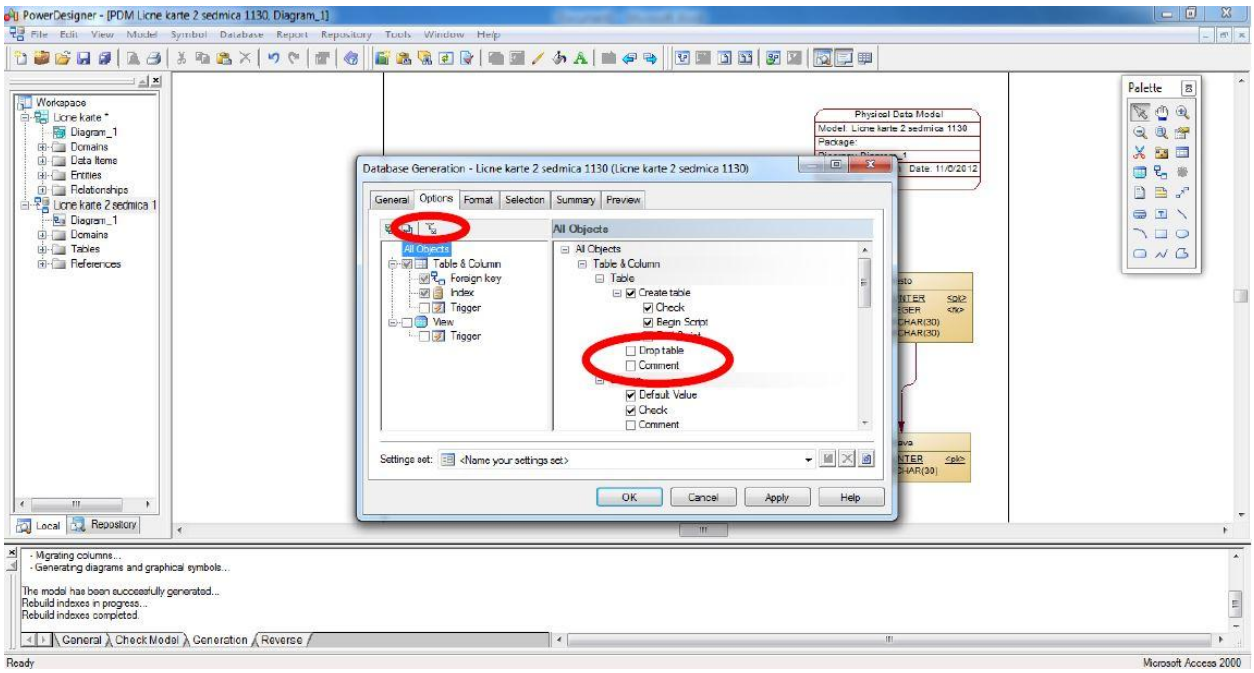
21. Креирање базе података у конкретном софтверу, мени Database, ставка Generate Database.



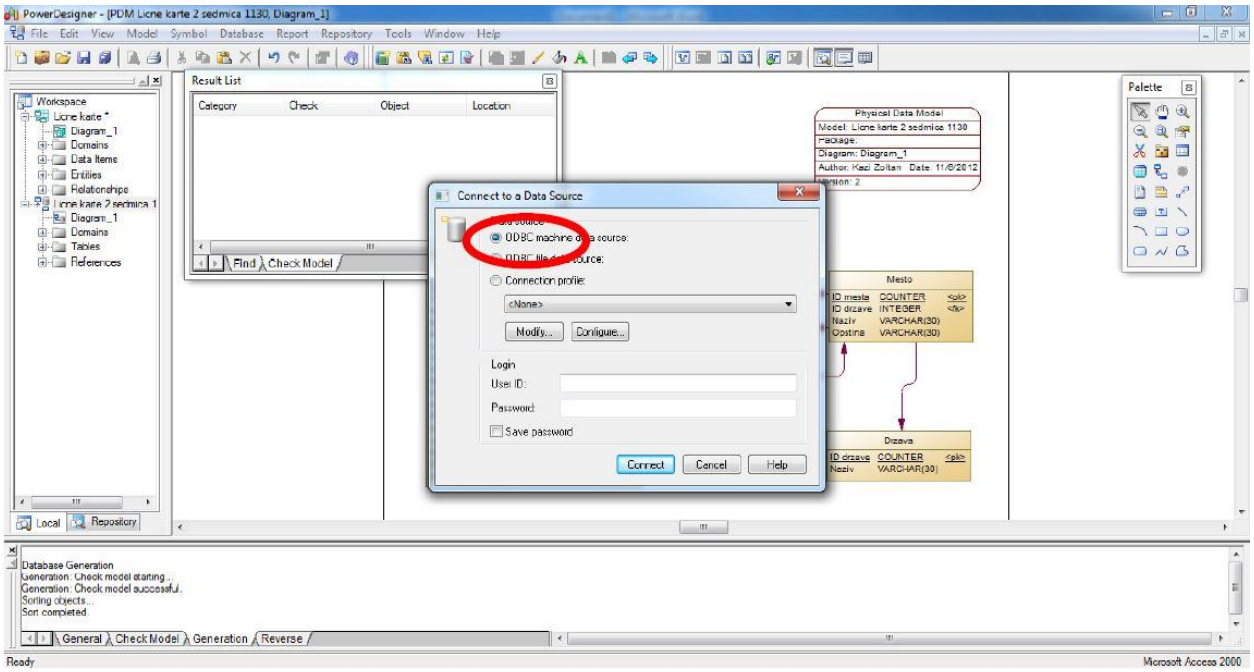
22. Креирање базе података, други корак, уписати/изабрати локацију и назив скрипт фајла са DDL SQL Л командама за креирања базе, који ће бити извршени преко ODBC механизма. Изабрати Direct generation.



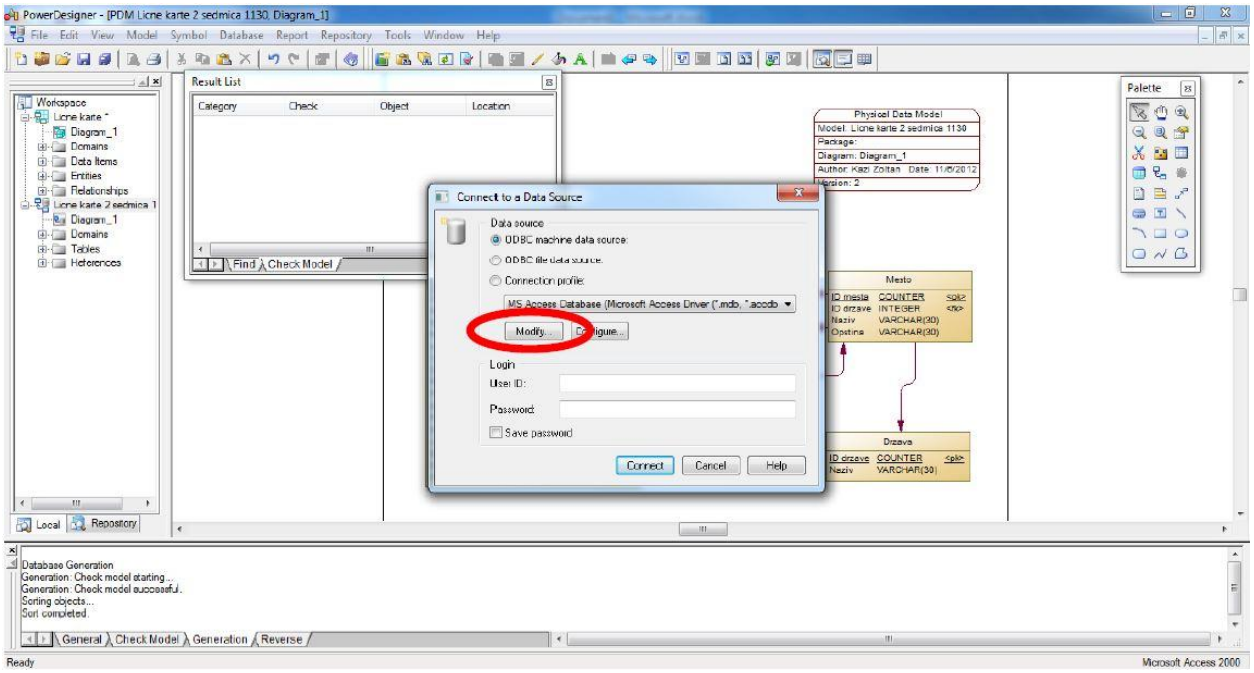
23. Креирање базе података, трећи корак - укључити сва подешавања, па затим команде DROP искључити када се први пут креира база података, иначе се приказују поруке са грешкама приликом извршавања SQL команди.



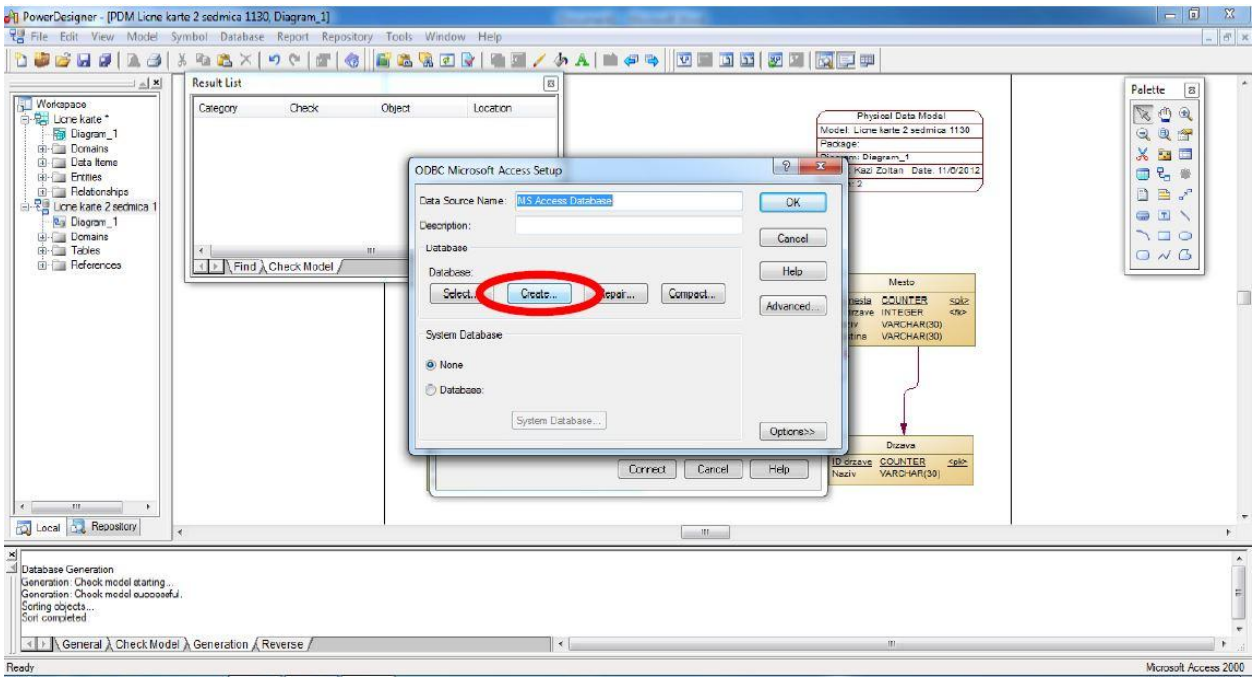
24. Креирање базе података, четврти корак – остваривање конекције до нове базе података, изабрати ODBC machine data source.



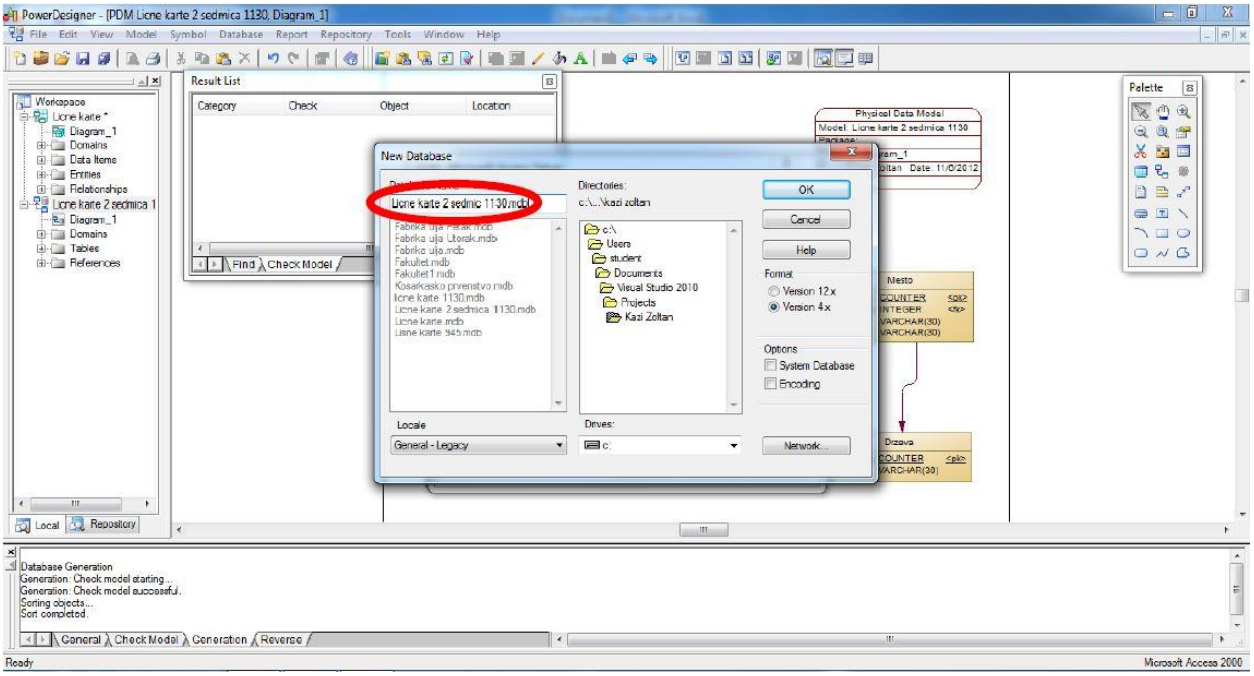
25. Креирање базе података, пети корак – избор одговарајућег драјвера, па затим следи опција Modify.



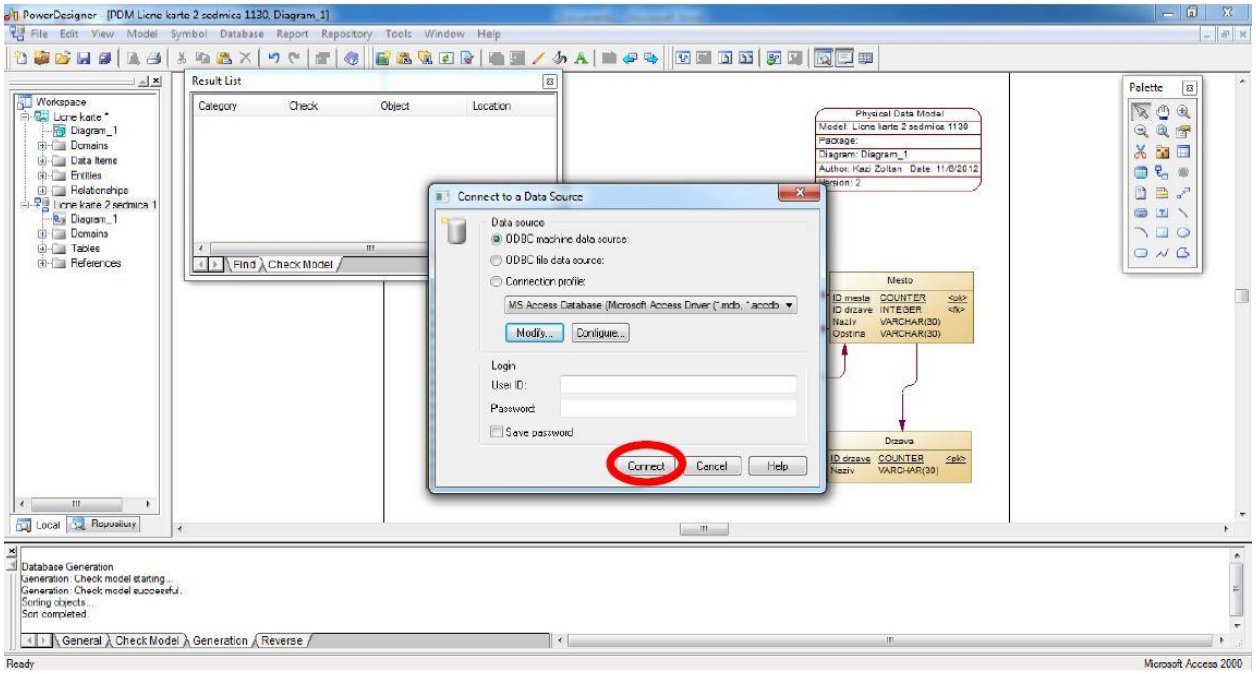
26. Креирање базе података, шести корак – креирање нове базе података у Access SUBP-у, изабрати тастер Create.



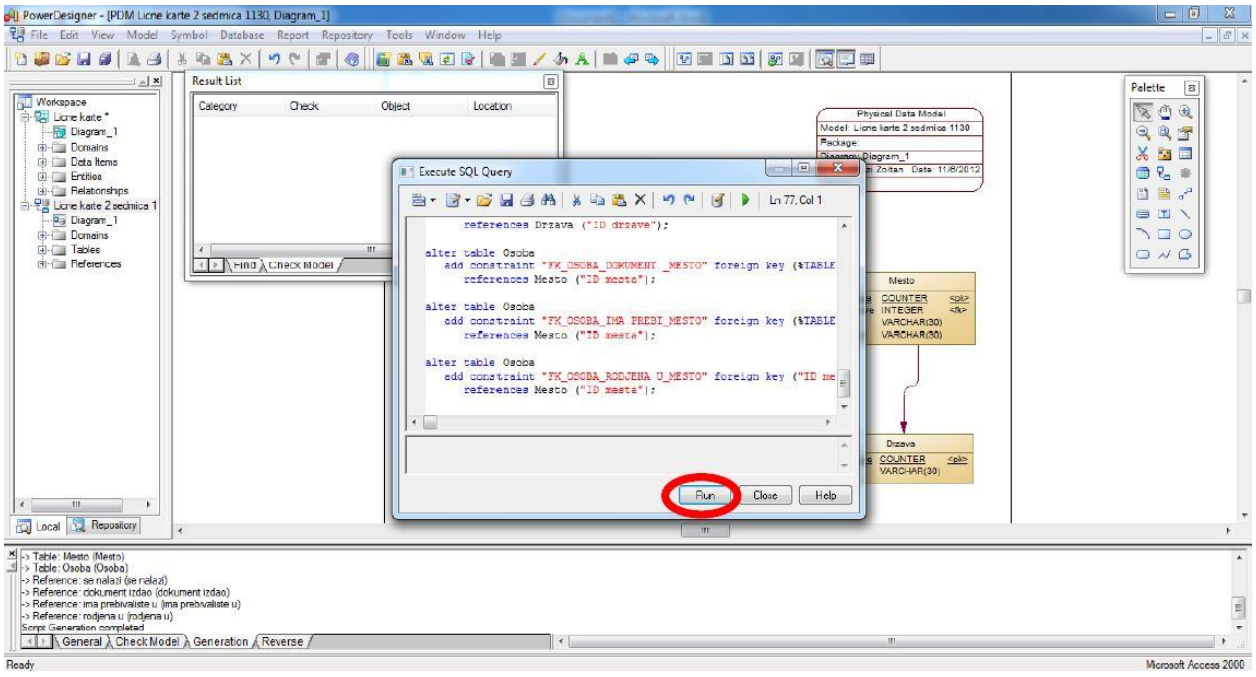
27. Креирање базе података, седми корак – изабрати локацију фајла базе и уписати назив базе (нпр. Личне карте.) и изабрати тастер ОК.



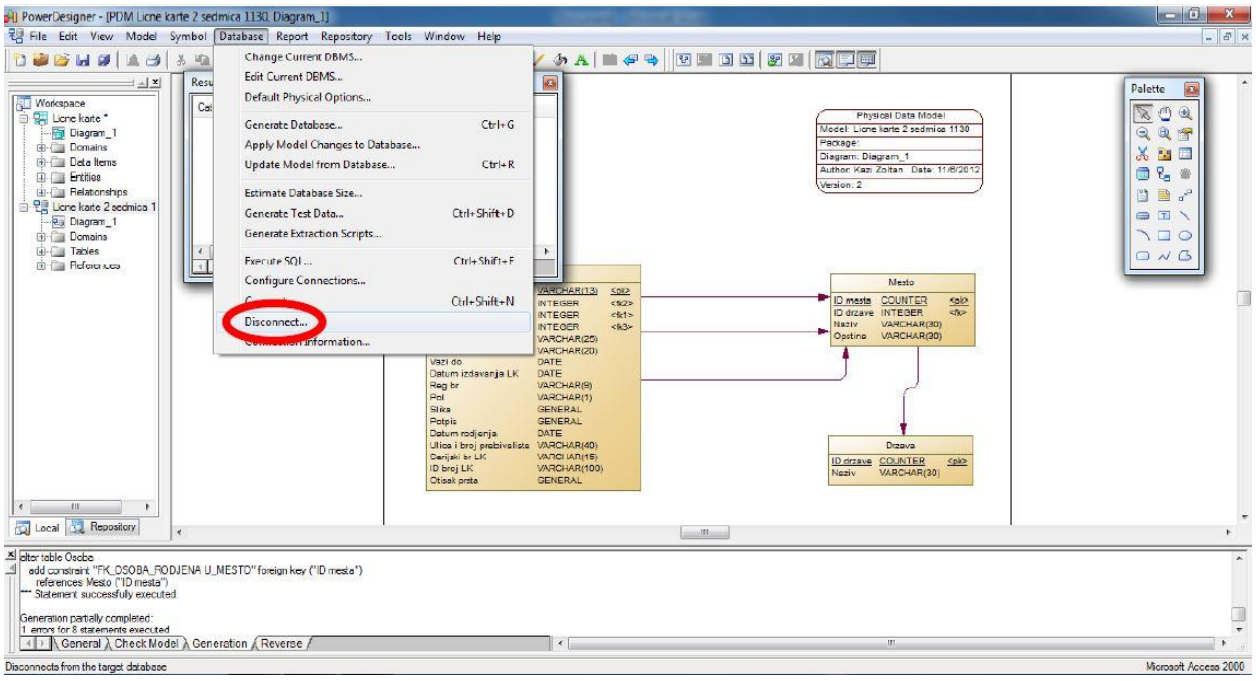
28. Креирање базе података, осми корак – изабрати тастер Connect како би се отворила конекција до новокреиране, празне базе.



29. Креирање базе података, приказ генерисаних SQL DDL комадни. Притиснути тастер Run. Уколико Power Designer пријављује грешке, бирати Ignore All, па проверити садржај прозора са порукама система како би се открили проблеми у извршавању SQL команди (Неопходно познавање SQL језика).



30. Након извршених SQL комадни, база података је генерисана. Затворити конекцију до базе преко менија Database-Disconnect и покренути MS Access софтвер и учитати фајл базе података.



5.0. ЗАКЉУЧАК

CASE алати представљају веома корисне специјализоване софтверске системе који омогућавају да се неки веома захтевни процеси код пројектовања база података аутоматизују и убрзају. Некада је било потребно ангажовати додатне особе које ће да раде документацију самог информационог система. Применом CASE алата могуће је аутоматски генерисати потребну документацију. Поред тога основна предности и карактеристике из приложеног су:

- Побољшана продуктивност;
- Побољшани квалитет;
- Смањено време одржавања;

И поред наведених предности CASE алати имају и неке осбине које ограничавају њихову примену. CASE алати не могу да замене креативно мишљење човека ,односно не могу да предвиде неке нелогичности у каснијем функционисању система. Можемо приметити да се нове генерације CASE алата константно усавршавају тако да се њихова примена software development-u константно увећава.

6.0. ЛИТЕРАТУРА

1. Вељовић Алемпије: Развој информационих система и базе података, Београд, 2003.

2. Драган Максимовић : Пројектовање информационих система, Факултет пословне економије, Бања Лука 2014.

3. Лазаревић Бранислав :Базе података, ФОН Београд, Београд 2003.

4. Велимир Милановић: DELPHI Базе података, Чачак 2012 .

5. http://www.infotrend.hr (Могућности CASE алата за моделирање пословних података - Томичић, Добровић, Видачић) ,2008.

6. http://office.microsoft.com