Базе података

Ненад Здравковић

Садржај:

I. Увод у базе података (БП) 3
1. Концепт база података 3
2. Организација података 4
3. Архитектура база података 5
4. Систем базе података 6
II. Релационе базе података 8
1. Модел података релационих база података 8
А) Структурни део модела 8
Б) Манипулативни део модела 9
В) Интегритетни део модела 9
2. Пројектовање (дизајн) базе података 10
III. Microsoft Access 22
1. Шта је то Access? 22
1. Креирање базе података испочетка 23
Отварање постојеће Access базе података 24
IV. Табеле 25
1. Увод у табеле 25
2. Креирање и рад са табелама 25
3. Својства табеле и поља 26
4. Типови података 26
5. Додавање поља у табелу 27
6. Постављање својстава поља 27
V. Увод у упите (Queries) 43
1. SQL (Structured Query Language 43
2. MS Access оператори 44
3. Креирање упита 45

I. Увод у базе података (БП)

Сви аспекти информационе науке и технологије у неком свом виду су везани за рад са подацима и информацијама. Услед тога концепт организације и управљања подацима од самог почетка па све до данашњих дана није изгубио ништа од почетног значаја и актуелности.

Податак је регистрована чињеница или догађај.
Информација је сазнање које добијамо из података. Информација је знање везано за ствари као што су чињенице, догађаји, концепти, објекти, идеје и процеси које унутар одређеног контекста имају одређено значење.

Управљање подацима обухвата:

- Контролу прикупљања, одабирања и складиштења података на медијумима;
- Контролу приступа подацима;
- Контролу коришћења података.

Информациони систем представља скуп интегрисаних компоненти за сакупљање, снимање, чување, обраду и преношење информација.

1. Концепт база података

Концепт база података је предмет истраживања од самог настанка информационих наука, са огромним утицајем како на економију тако и на друштво у готово свим аспектима живота.
Базе података представљају основу сваког иоле озбиљнијег информационог система и фундаментално мењају начин на који организације функционишу.
Значај база података је додатно повећан последњих година са значајним развојем самог хардвера, комуникација, Интернета, електронске трговине, бизнис интелигенције и мобилне комуникације.

Неке од најрепрезентативнијих области употребе:

1) Привредни системи:
   a) Продаја (клијенти, производи, наруџбине)
   b) Рачуноводство (плаћања, билансна стања, рачуни, инвентар)
   c) Људски ресурси (запослени, плате, порези и доприноси)
   d) Производња (набавни канали, праћење процеса производње, залихе и поруџбине)
2) Банкарство и финансије
   a) Банкарство (клијенти, рачуни, зајмови и банкарске трансакције)
   b) Кредитне карте (куповине путем картица, месечни биланси стања)
   c) Финансије (информације о компанији, продаји и набавци, о тржишту и сл.)
3) Универзитети (информације о студентима, смеровима, оценама)
4) Авио компаније (резервације, распоредима)
5) Телекомуникације (позиви, месечни рачуни, биланси на припејд картицама, информације о комуникационим мрежама)

Базе су организоване колекције информација. База података представља колекцију података који се односе на одређену тему или циљ.

База података представља скуп међусобно повезаних података о неком реалном систему који се описује.

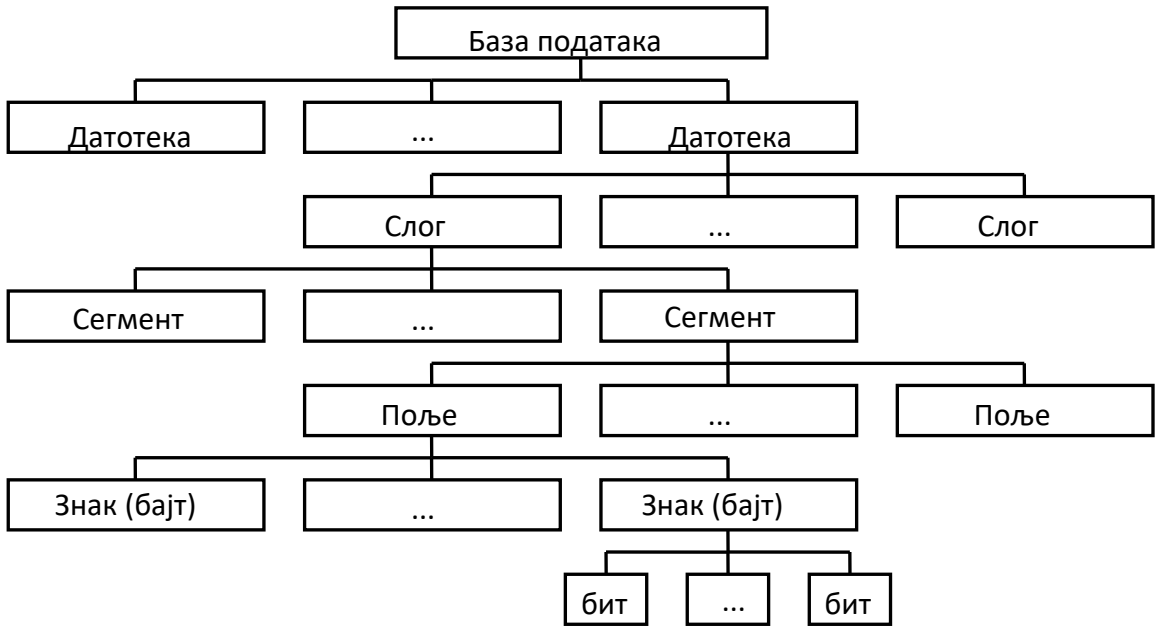
Састоји се од објеката као што су табеле, упити, формулари, извештаји.
За разлику од класичног, конвенционалног приступа организацији података преко датотека, код база података имамо да су структура и садржај података независни од процеса који их користе. Код добро пројектованих база података обезбеђује се потпуна логичка и физичка независност података.

2. Организација података

Организација података треба да олакша управљање подацима и њихово коришћење.
Подаци се групишу у организационе целине, где разликујемо следеће јединице:

- **Бит** (бинарна цифра 0 или 1) представља најмању количину информација која се може изразити. Бит се не може адресирати, па самим тим ни самостално користитити.
- **Бајт** (byte) представља најмању адресибилну јединицу података (може јој се директно приступити), и састоји се од 8 бита.
- **Поље** представља групу од 2 или више знака. Можемо га схватити као меморијски простор у ком се може регистровати конкретна вредност податка. Податак, односно поље означавају се именом податка, односно именом поља. Тако име поља Град може имати следеће вредности: Београд, Краљево, Суботица и сл. Садржај поља могу бити разни подаци: бројни, текстуални и др. Број знакова одређује његову дужину, а она се одређује према највећем податку који може бити садржан у пољу.
- **Сегмент** или групно поље се састоји од два или више поља над којима се могу вршити логичке операције истовремено. Пример је датум који се састоји од дана, месеца и године. Некада је потребно да се узме вредност појединог поља посебно, а понекад вредност садржаја групног поља - сегмента. Број и дужина поља одређује величину сегмента.
- **Слог** или запис чини више сегмената или поља чији се садржај односи на један појам, појаву или догађај, а који су заједно доступни за обраду. Структуру слога одређују: број, врста и дужина поља. Дужина слога одређује се бројем и дужином поља од којих се састоји. Може бити фиксне или променљиве дужине. Да би се сваки слог идентификовао између више слогова потребно је да има кључ. Уколико има више кључева један је примарни, а остали су секундарни.
- **Датотека** чини скуп слогова, и садржи по неком критеријуму сродне податке, који се чувају под јединственим именом на неком виду спољне меморије. Она је највећа организациона јединица у класичној организацији података. Број и дужина слогова одређују величину датотеке.
- **База података** је састављена од више међусобно повезаних датотека.

Да би се управљало подацима потребно их је претходно стуктуирати прецизним правилима.



Дијаграм 1. Организационе јединице података

Структура се дефинише скупом елемената и релацијама између елемената.

У зависности од тога над којим се елементима дефинишу структуре података се деле на:

- Логичке структуре података (ЛСП);
- Физичке структуре података (ФСП).

Обе структуре се у општем случају дефинишу помоћу релација над скупом садржаја слогова истих или различитих врста. Разлика је у томе што ЛСП представљају садржаје слогова са становишта како их користи програмер, а ФСП са становишта како су они регистровани на неком физичком медијуму. Једној ЛСП може одговарати више ФСП, док једној ФСП може одговарати само једна ЛСП.

3. Архитектура база података

Архитектура система база података ради се према предлогу тима америчког националног института за стандарде познатог као ANSI/X3/SPARC модел, на тај начин што се подаци третирају у три одвојена нивоа, где сваки ниво хијерархије укључује специфичан начин представљања, репрезентацију објеката, односа међу објектима и операција над објектима.

ANSI архитектура обухвата следеће хијерархијске нивое:

- **Унутрашњи (интерни) ниво** је најнижи ниво који представља физичку дефиницију података, њихову физичку структуру и начин приступа подацима на медијумима.
- **Концептуални или средњи ниво** дефинише уопштен модел података како је "виде" (користе) сви корисници.
- **Спољашњи (екстерни) ниво** је највиши ниво који дефинише како поједини специфични корисници "виде" (користе) податке независно од тога где и како су они физички регистровани.

Унутрашњи модел дефинише место и начин регистровања података на медијумима и он представља **физички модел базе података**.

Спољашњим и концептуалним моделом дефинише се садржај базе података и они представљају **логички модел базе података**.

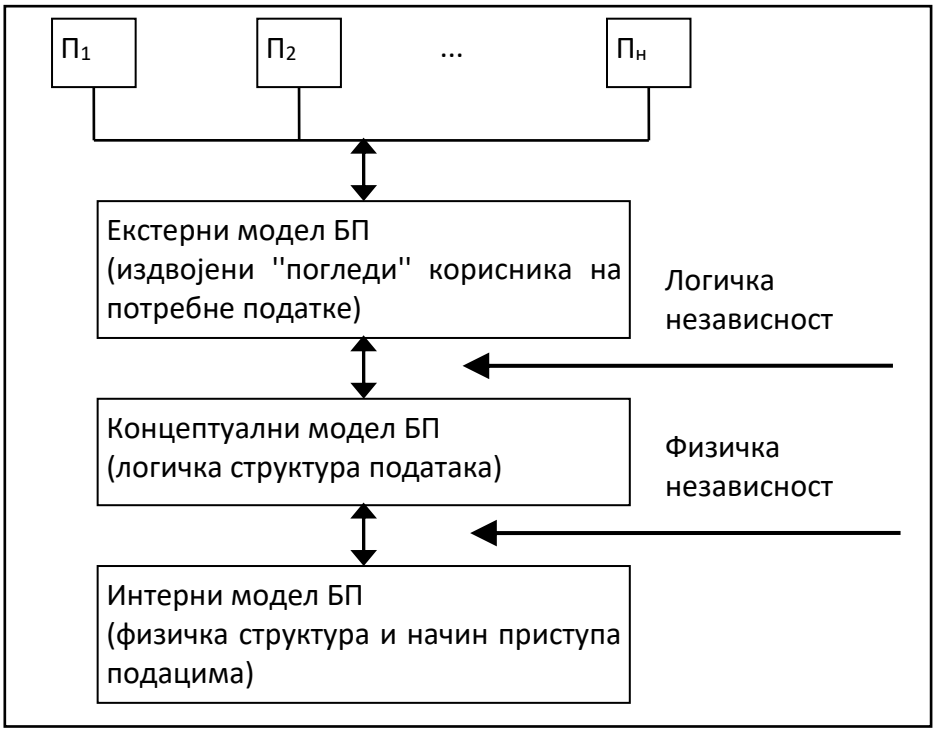
Логичким моделом представљају се и подаци и релације између њих, као и скуп операција које корисник може да изврши над подацима. И као и реални систем он је пре динамички него статички.

На концептуалном нивоу се могу "видети" сви подаци из физичке базе података, само што је њихова репрезентација погоднија од физичке, јер је на вишем нивоу апстракције. Репрезентација на овом нивоу се зове **модел података**. Спољашњи ниво даје појединим корисницима могућност да имају сопствена гледања на модел података, па се погледи (подмодели) налазе на највишем нивоу апстракције. На овом нивоу се неки подаци могу приказати, док се други могу сакрити.

Зато постоји тачно један модел података у систему, који се односи на целокупност базе података, и већи број спољашњих погледа, од којих се сваки састоји од апстрактне слике дела базе података.

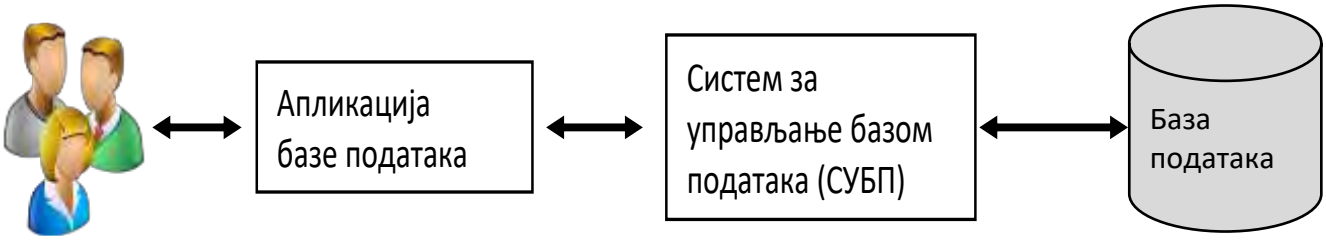
- **Логичка независност података** значи да измена концептуалног модела података (додавање и брисање поља, успостављање нових веза и сл.), не утиче на корисничке програме који користе податке.
- **Физичка независност података** значи да измене физичке структуре података не доводе до измене концептуалног и екстерног модела, као и самих корисничких програма. То значи да се може заменити медијум или извршити реорганизација података на уређају без промене готовог апликативног програма.

Дијаграм 2. Шема логичке и физичке независности података



4. Систем базе података

Систем базе података је један комплексни систем код кога разликујемо четири компоненте:
(1) База података (2) Систем за управљање базама података [ЦУБП] (3) Апликација базе података (4) Корисници



Дијаграм 3. Систем базе података

Више корисника преко више апликација може да приступи једној бази података.
Најважнији подсистем овде је систем за управљање базом. Систем за управљање базом података је компјутерски заснован систем за управљање и складиштење међусобно повезаних података који даје интерфејс за приступ подацима. Он омогућава рад са великом количином података на прикладан и ефикасан начин.

Систем за управљање базом података је софтверски систем који омогућава дефинисање, креирање, одржавање и контролу приступа бази података.

Неки од најпознатијих СУБП система:

- Microsoft Access
- Microsoft SQL Server
- Oracle Database
- Oracle MySQL
- IBM DB2

Систем за управљање базом података (СУБП) омогућава:

- **Контролу редундантности података.** Потребно је што је више могуће редуковати и минимизирати дуплирање података. Тиме се ефикасније користи меморија, али што је још важније смањује вероватноћа настанка неконзистентности података. Неконзистентност података се јавља када се исти податак налази на два или више места у бази података, а изврши се измена на само једном месту. Неконзистентност доводи до нарушавања интегритета базе података. Зато сваки податак треба да се чува на само једном месту.
- **Одржавање интегритета података.** Подаци који се чувају у бази морају да буду тачни. Нетачност података може да се појави услед претходно поменуте неконзистентности података, или услед непоштовања неких валидних ограничења (нпр. 30. Фебруар би био нетачан податак). Зато подаци морају да задовоље одређена ограничења, чиме се постиже њихова тачност.
- **Омогућавање лаког приступа подацима.** Ово је једна од главних особина свих система за управљање базама података. Сами подаци су бескорисни уколико не може да им се приступи брзо и лако ради њиховог даљег процесирања, и СУБП пружа добре алате и интерфејс за ту сврху.
- **Обезбеђивање сигурности података.** СУБП даје подршку за више корисника, где обично нема сваки корисник приступ свим подацима. СУБП даје методе за рестрикцију приступа корисницима и побољшавање сигурности података.
- **Контрола паралелног и истовременог приступа подацима.** Ова функција такође проистиче из ситуације где више корисника може да користи базу у исто време. Уколико два корисника одлуче да промене исти податак са различитим вредностима, резултат би био нетачност података. Такође би било непријатно уколико би рецимо на аеродрому, два службеника покушала да продају исто место за неку дестинацију. СУБП даје могућност решавања проблема овог типа.
- **Омогућавање дељења података.** СУБП омогућава да се један податак дели између различитих апликација.

II. Релационе базе података

Већина система за управљање базама података доступних данас је базирана на релационом моделу. Релациона база података је база где су подаци организовани у међусобно повезане табеле. Табеле имају задатак да смислено групишу податке о истом објекту, који представља репрезентацију неког стварног објекта из реалног система који се описује. Табеле се састоје од записа (редова) и поља (колона). Релациона база података се тако састоји од самих података, као и релација између њих. Први релациони модел база података представљен је раних 1970-тих у IBM San Jose истраживачком центру од стране E.F.Codd-а. Овај модел је дизајниран према скупу математичких концепата познатих као "**релациона алгебра**".

1. Модел података релационих база података

Модел података (логички модел) чини:

- **Структурни** део модела - скуп типова објеката;
- **Манипулативни** део модела - скуп операција над објектима;
- **Интегритетни** део модела - скуп правила интегритета.

А) Структурни део модела

Структурни део модела се састоји од:

✓ **Ентитета** - објекат са скупом повезаних карактеристика које га једнозначно идентификују и чине једну целину. Уколико објекат има више повезаних карактеристика, онда је он ентитет. Тако Поруџбина има рецимо атрибуте као што су статус, идентификатор (коме је нешто продато) и датум поруџбине, који чине Поруџбину ентитетом.

✓ **Атрибута** - тј. особине, својства, или карактеристике ентитета. Уколико објекат има само једну карактеристику или вредност, онда је он атрибут. Тако БројПоруџбине идентификује број поруџбине и ништа више, тако да не може да буде ентитет.

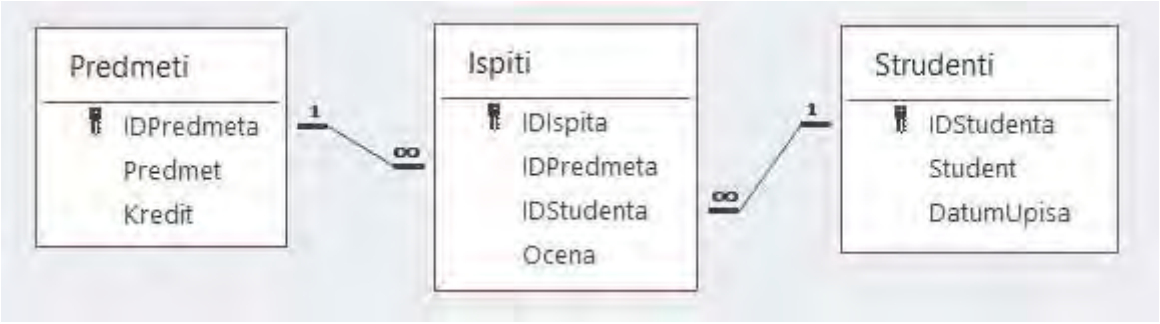
✓ **Релација** - тј. асоцијативна веза између два ентитета. Уколико опис објекта садржи глагол, онда је он вероватно релација. Тако на пример Купац захтева Поруџбину описује релацију између ентитета Купац и ентитета Поруџбина.

Било која два ентитета могу да имају једну од следећих типова релација:

- **"Један-према-један"**: А је асоциран са само једном вредношћу Б и обрнуто;
- **"Један-према-више"**: А може бити асоциран са много вредности Б, док је са друге стране Б асоциран са само једном вредности А;
- **"Више-према-више"**: А је асоциран са многим вредностима Б и обрнуто Б је асоциран са много вредности А.

Концепт релационих база података демонстрираћемо на једном малом примеру замишљене БП једног факултета. Издвојили смо три ентитета и креирали на основу тога три табеле: **Предмети**, **Студенти** и **Испити**. Од релација дефинисали смо један према више између ентитета **Студент** и **Испит**, као и релацију један према више између ентитета **Предмет** и **Испит**. Видимо да студент може да полаже више испита, а један предмет такође може да има више испита. Уочићемо да између ентитета **Предмет** и **Студент** постоји веза више према више, успостављене на индиректан начин преко ентитета Испити, што фактички значи да један предмет може да полаже више студената, а да један студент може исто тако да полаже више предмета.

Овај дизајн је могуће видети на следећем дијаграму 4:



Б) Манипулативни део модела

Манипулативни део модела се састоји од скупа операција. Операцијама се задају упити и радње над ентитетима, као што су претраживање, филтрирање или мењање (ажурирање) података о ентитетима. Тако рецимо у систему за вођење издавачке делатности могли би да имамо следеће операције:

- регистровање нове књиге
- расходовање књиге
- измена статуса издавача

В) Интегритетни део модела

Интегритетни део модела са скупом правила интегритета који дефинишу својства података и односе између података, који морају бити задовољени да би стање базе података било исправно. Разликујемо следеће врсте интегритета:

- **Интегритет ентитета** – где сваки запис (ред) мора да буде јединствен примерак (инстанца) ентитета који га садржи. Ово се постиже преко креирања поља за примарни кључ, које би представљало јединствени индекс записа.
- **Интегритет домена** – који осигурава исправност актуелних вредности података. Домен представља скуп величина из кога поља (атрибути) могу да узму конкретну вредност. Ова врста интегритета се постиже преко одређивања типова података, одређивања да ли поље може да има нулл (непознату) вредност, као и одређених правила базе података.
- **Референцијални интегритет** – који се користи да би се осигурало да страни кључ коректно одговара примарном кључу у родитељској (примарној) табели. Страни кључ је поље у зависној (дете) табели за који се скуп могућих вредности може наћи у примарном кључу у примарној табели, и преко кога се успоставља релација између две табеле. Овај тип интегритета се такође користи за управљање каскадним променама од родитељских ентитета према деца ентитетима.

Основни концепти релационих база података обухватају:

- **Сваки ентитет има скуп атрибута који јединствено дефинишу примерак тог ентитета** - овакав скуп атрибута се назива примарни кључ. Примарни кључ може бити састављен од једног или више атрибута.
- **Ниједан део примарног кључа не може да има null (непознату) вредност** - пошто је првенствена намена примарног кључа да јединствено идентификује сваки запис ентитета, са нулл вредношћу не би могло наравно ништа да се идентификује.
- **Табела не може да има дуплиране редове** - ово је последица тога да свака табела мора да има примарни кључ, који не дозвољава дуплирање редова. Ово представља као што је речено интегритет ентитета или података.
- **Табеле су међусобно повезане** - ово повезивање се остварује преко страног кључа и представља референцијални интегритет.
- **Редослед редова у табели је произвољан** - једини начин да се зна редослед редова је да се изричито одреди.
- **Редослед колона у табели је произвољан** - при отварању табеле овај редослед је једнак оном како су колоне формиране, али се преко упита тај редослед може диктирати по жељи.

2. Пројектовање (дизајн) базе података

Исправно дизајнирана база података нам омогућава приступ ажурним и тачним информацијама. Пошто је исправан дизајн од суштинске важности за постизање циљева у раду са базом података, улагање времена у стицање знања о принципима доброг дизајна има смисла. На крају крајева, у том случају много је већа вероватноћа да ћемо добити базу података која одговара нашим потребама и која на једноставан начин може да се мења.

Пројектовање или моделирање релационе базе података обухвата четири главне фазе:

1. **Логички дизајн базе података** - где се идентификују реални објекти - ствари, људи и сл. који се затим претварају у ентитете, логичке објекте са својим атрибутима за сваки важан део информација везан за саме ентитете. Ентитети се затим повезују међусобно преко релација.
2. **Физички дизајн базе података** - где се идентификовани објекти из претходне фазе, ентитети пребацују у физичке објекте - табеле, а атрибути у колоне. Релације између ентитета се успостављају преко примарних и страних кључева који морају да се дефинишу.
3. **Нормализацију података**.
4. **Креирање базе података и самих табела** - најчешће преко упита, тј. одговарајућег SQL кода.

Процесом дизајнирања базе података управљају следећи принципи:

Први принцип је да су дуплиране информације (које се називају и редундантни подаци) лоша појава, јер непотребно заузимају простор и повећавају вероватноћу појаве грешака и недоследности.

Други принцип истиче важност исправности и потпуности информација. Ако база података садржи неисправне информације, извештаји који повлаче информације из базе података ће такође имати те исте неисправне информације. Као последица тога, донете одлуке које су засноване на тим извештајима ће у основи бити погрешне или неадекватне.

Потребно је да дизајн одговара нашим потребама у вези са обрадом података и извештавањем.

Основни кораци приликом дизајнирања базе:

1. Одређивање сврхе базе података
2. Одређивање потребних поља
3. Одређивање табела
4. Одређивање која поља припадају одређеним табелама
5. Одређивање поља (или групе поља) која једнозначно идентификују сваки запис
6. Одређивање релација између табела
7. Прочишћавање дизајна базе
8. Побољшање дизајна применом правила нормализације
9. Креирање осталих објеката базе података и уношење података

Дизајн базе података, корак по корак на примеру базе података за продају и пословање:

1. Одређивање сврхе базе података

Овај корак доприноси припреми за преостале кораке.
Препоручује се да запишете намену базе података на папир – саму намену, начин на који планирате да је користите, као и ко ће је користити. На пример, за малу базу података за посао који радите код куће могли бисте да запишете нешто једноставно попут *„База података купаца садржи листу информација о купцима са наменом креирања пошиљки и извештаја“*. Ако је база података сложенија или је користи већи број људи, опис намене ће бити обимнији, а требало би такође да садржи време када ће и начин на који ће свака особа користити ту базу података.

Неке од основних активности у овој фази тако укључују:

- Разговор са људима који ће користити базу
- Скицирање извештаја које база треба да произведе
- Прикупљање образаца који се тренутно користе за прикупљање података

Одређивањем сврхе, појавиће се листа информација везаних за базу. Одредиће се које чињенице база треба да чува и ком објекту припадају те чињенице.

Ове чињенице одговарају пољима (колонама), а објекти одговарају табелама.

2. Проналажење и организовање потребних информација или поља

Свако поље представља одређену информацију о одређеном објекту или теми.

Том приликом треба водити рачуна о следећим принципима:

- Укључити све информације које ће бити потребне
- Информације треба чувати у најмањим логичким целинама. Тако на пример име треба раздвојити у два поља: Име и Презиме, како би подаци били лако сортирани по презимену.
- Поље не сме да има листу података
- Не укључивати поље које се изводи или рачуна
- Не правити поља која су међусобно слична: Производ1, Производ2, Производ3 ...

Увек треба поћи од постојећих информација. На пример, можда излазне поруџбине записујете у регистратор или информације о купцима чувате на папирним обрасцима у архиви. Сакупите те документе и наведите све приказане типове информација (на пример, свако поље које попуњавате у обрасцу). Ако немате постојеће обрасце, замислите да морате да дизајнирате образац како бисте записали информације о купцима. Које информације бисте унели у образац? Која поља за попуњавање бисте креирали? Идентификујте и наведите све те ставке.
На пример, претпоставите да тренутно листу купаца чувате на картицама за индексирање. Прегледање ових картица би могло да покаже да свака картица садржи име, адресу, град, државу, поштански број и број телефона купца. Свака од ових ставки представља потенцијалну колону у табели.

Док припремате листу, немојте се трудити да она одмах буде савршена, већ наведите сваку ставку које се сетите. Ако ће још неке особе користити базу података, и њих упитајте за идеје. Касније можете да прецизније подесите листу.

Након тога размотрите типове извештаја и пошиљки које ћете можда желети да креирате на основу базе података. На пример, можда бисте желели да креирате извештај о продаји производа који ће

продају приказивати по региону или извештај са резимеом залиха који приказује нивое залиха производа. Можда бисте желели да генеришете и типска писма која ћете слати купцима најављујући продајни догађај или нудећи премију.

Дизајнирајте извештај у глави и замислите како ће изгледати. Које информације бисте обухватили извештајем? Наведите сваку ставку. То исто учините за типско писмо, као и за сваки други извештај чије креирање планирате.

Размишљање о извештајима и пошиљкама које бисте желели да креирате помаже вам да идентификујете ставке које ће вам бити потребне у бази података. На пример, претпоставимо да сте купцима омогућили да дају (или одбију) сагласност за периодичне исправке које се шаљу путем е-поште и да желите да одштампате листу оних који су дали сагласност. У табелу са купцима додајте колону „**Slanje e-poste**“ да бисте записали те информације. Поље за сваког купца можете да поставите на вредност „Да“ или „Не“.

Потреба да се купцима шаљу е-поруке подразумева још једну ставку коју је потребно записати. Када сазнате да купац жели да прима е-поруке, мораћете да знате и е-адресу на коју ћете их послати. Стога је потребно да запишете е-адресу сваког купца.

Има смисла креирати прототип сваког извештаја или излазне листе и размотрити које ставке ће бити потребне за креирање извештаја. На пример, када прегледате типско писмо, присетићете се неколико ствари. Ако желите да укључите одговарајуће ословљавање – на пример, префикс „г.“, „гђа“ или „гђица“ којом започиње поздрав, мораћете да креирате и ставку ословљавања. Осим тога, могли бисте да писмо започнете на уобичајен начин са „Поштовани г. Савић“, а не „Поштовани г. Светозар Савић“. Ово би значило да ви, у принципу, желите да презиме ускладиштите одвојено од имена.

Кључна ствар коју би требало да имате у виду је да сваку информацију треба да рашчланите на најмање корисне делове. Кад су у питању имена, пуно име поделите на два дела – име и презиме, како би презиме било одмах доступно. Одвојено складиштење презимена купаца може да буде корисно ако, на пример, извештај желите да сортирате по презимену. Уопште узевши, ако желите да сортирате, претражујете, израчунавате или извештавате на основу неке ставке са информацијама, требало би да ту ставку сместите у посебно поље.

Размислите о питањима на која можда желите да база података пружи одговор. На пример, колико сте продаја препорученог производа закључили прошлог месеца? Где живе ваши најважнији купци? Ко је добављач вашег најпродаванијег производа? Уколико предвидите ове одговоре, бићете у могућности да се усредсредите на додатне ставке које треба да запишете.

После сакупљања ових информација спремни сте за следећи корак.

3. Одређивање табела

Свака табела треба да садржи информације о само једном објекту или теми. Листа поља даће идеју о којим табелама (објектима) је реч. Одаберите главне ентитете или теме како бисте информације распоредили у табеле. На пример, пошто пронађете и организујете информације за базу података продаје производа, прелиминарна листа би могла да изгледа овако:

Главни ентитети које су овде приказани су Производи, Добављачи, Купци и Поруџбине.

Стога, креирање има смисла почети са ове четири табеле: једна је за чињенице о производима, друга за чињенице о добављачима, трећа за чињенице о купцима, а четврта за чињенице о поруџбинама.

Иако овим листа није довршена, то је добар почетак. Можете да наставите с прочишћавањем ове листе све док не добијете дизајн који добро функционише.

Када први пут будете кориговали прелиминарну листу ставки, можда ћете пожелети да их сместите у једну табелу, уместо у четири које су приказане на претходној слици. Сада ћете сазнати зашто би то било лоше. Накратко размотрите табелу која је овде приказана:

Proizvodi i dobavljači

Ime proizvoda	Dobavljači	Adresa
Čaj	Egzotični napici	Ulica Gandijeva
Rakija od pirinča	Egzotični napici	Ulica Gandijeva
Sirup od anisa	Egzotični napici	Ulica Gandijeva
Chef Anton's Kejdžun za	New Orleans Kejdžun sp	Poštanski fah 78

У овом случају, сваки ред садржи информације и о производима и о добављачима. Пошто можете имати велики број производа од истог добављача, информације о имену добављача и његовој адреси морају се поновити више пута. На тај начин се заузима непотребан простор на диску. Много је боља опција да само једном запишете информације о добављачу у одвојену табелу „**Dobavljaci**“, а затим је повежете са табелом „**Proizvodi**“.

Други проблем са дизајном у овом случају јавља се када је потребно да измените информације о добављачу. На пример, претпоставимо да треба да промените адресу добављача. Пошто се она појављује на много места, може да се деси да случајно промените адресу на једном месту, а да заборавите да је промените на осталим местима. Записивање адресе добављача на само једном месту решава тај проблем.

Када дизајнирате базу података, увек покушајте да сваку чињеницу запишете само једном. Ако је потребно да исту информацију, попут адресе одређеног добављача, понављате на више места, сместите ту информацију у одвојену табелу.

На крају, претпоставимо да постоји само један производ винарије „**Coxo Vinery**“, а ви желите да га избришете без губљења информација о имену и адреси добављача. Како ћете избрисати запис о производу, а да при томе не изгубите и информације о добављачу? То није могуће. Пошто сваки запис садржи чињенице о производу, као и чињенице о добављачу, не можете да избришете једно, а да не избришете и друго. Да бисте ове чињенице чували одвојено, табелу морате поделити на два дела: једна табела служи за информације о производу, а друга за информације о добављачима. Брисање записа о производу би требало да избрише чињенице о производу, а не и чињенице о добављачу.

Када одаберете тему за табелу, колоне у тој табели би требало да складиште само чињенице о теми. На пример, табела производа би требало да складишти само чињенице о производима. Пошто је адреса добављача чињеница о добављачу, а не чињеница о производу, она припада табели о добављачима.

4. Распоређивање поља по одређеним табелама

Одлучите које информације треба да пратите у вези са темом записаном у табели како бисте одредили колоне у табели. На пример, колоне „**Ime**“, „**Adresa**“, „**Grad-drzava-postanski broj**“, „**Slanje e-poste**“, „**Oslovljavanje**“ и „**E-adresa**“ представљају добру почетну листу колона за табелу „**Kupci**“. Сваки запис у табели садржи исти скуп колона тако да за сваки запис можете да ускладиштите информације о имену, адреси, граду-држави-поштанском броју, слању е-поште, ослобљавању и е-адреси. На пример, колона са адресама садржи адресе купаца. Сваки запис обухвата податке о једном купцу, а поље за адресу садржи адресу тог купца.

Након што одредите почетни скуп колона за сваку табелу, колоне можете додатно да прочистите. На пример, добро је ускладиштити пуно име купца у две одвојене колоне – име и презиме, како бисте могли да сортирате, претражујете и индексирате само према овим колонама. Слично томе, адреса се у ствари састоји од пет одвојених компоненти – адресе, града, државе, поштанског броја и земље/региона, које је корисно ускладиштити у одвојене колоне. На пример, ако желите да извршите операцију претраживања, филтрирања или сортирања по држави, потребна вам је информација о држави која је ускладиштена у одвојеној колони.

Требало би да размотрите да ли ће база података имати само информације домаћег порекла или и међународног. На пример, ако планирате да складиштите међународне адресе, боље је уместо колоне „**Drzava**“ имати колону „**Region**“ јер таква колона може да прихвати и домаће државе, као и регионе других земаља/региона. Слично томе, колона „**Postanski broj**“ има више смисла него колона „**Postanski broj – Srbija**“ ако ћете складиштити међународне адресе.

Приликом одлучивања која поља ће припадати одређеним табелама, треба имати на уму следеће принципе:

- **Додати поље само једној табели**
- **Не додавати поље ако ће резултирати да се иста информација појављује у више записа**.

Уколико је тако, то поље вероватно припада погрешној табели.

- **Не укључивати израчунате податке**

У већини случајева, резултат рачунских операција не би требало да складиштите у табеле. Податке који се рачунају можемо добити у каснијој фази коришћења базе, онда кад нам је то потребно.

- **Информације ускладиштите у најмање логичке јединице**

Можда ћете пожелети да имате једно поље за пуна лична имена или имена производа са описима производа. Ако комбинујете више врста информација у једном пољу, касније ће бити тешко преузети појединачне чињенице. Потрудите се да информације рашчланите на логичке јединице; на пример, креирајте одвојена поља за име и презиме или име, категорију и опис производа.

Када прочистите колоне са подацима у свим табелама, моћи ћете да одаберете примарни кључ за сваку табелу, што је главна активност следећег корака.

5. Одређивање поља (или групе поља) која чине примарни кључ

Свака табела би требало да садржи колону или скуп колона који јединствено идентификују сваки ред ускладиштен у табели. То је често јединствени идентификациони број, попут ID броја запосленог или серијског броја. У терминологији базе података ова информација се назива **примарни кључ** табеле.

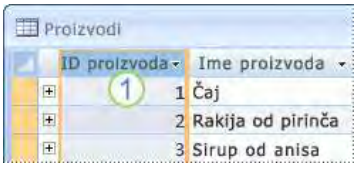
Ако већ имате јединствени идентификатор табеле, као што је број производа који јединствено идентификује сваки производ у каталогу, тај идентификатор можете да користите као примарни кључ табеле – али само ако ће вредности за сваки запис у овој колони увек бити различите.

За неке табеле имаћемо расположива поља која представљају природне примарне кључеве. Пример би био за ентитет Књиге такозвани ISBN (International Standard Book Number) број чија је 13-то цифарна шифра јединствена за сваку књигу, или ЈМБГ (Јединствени матични број грађанина) који је такође јединствен и исто тако се састоји од комбинације 13 цифара. То би било у реду ако би све особе које се воде били држављани наше земље, међутим ако би водили и стране држављане, онда ЈМБГ не би више могао да буде прикладан у таквом сценарију.

У примарном кључу не можете имати дуплиране вредности. На пример, имена особа немојте користити као примарни кључ јер она нису јединствена. Лако би могло да се деси да две особе у једној табели имају исто име. Такође, примарни кључ увек мора да има вредност. Ако постоји могућност да вредност колоне у неком тренутку постане недодељена или непозната (вредност која недостаје), она не може да се користи као компонента у примарном кључу.

Увек би требало да одаберете примарни кључ чија се вредност неће мењати. У бази података са више табела примарни кључ једне табеле се може користити као референца у другим табелама. Ако се примарни кључ промени, промена се мора применити и на свим местима на којима се референцира тај кључ. Коришћење примарног кључа који се неће мењати смањује вероватноћу да примарни кључ постане несинхронизован са другим повезаним табелама.

Као примарни кључ се често користи произвољни јединствен број. На пример, свакој поруџбини можете да доделите јединствени број поруџбине. Једина намена броја поруџбине је идентификација поруџбине. Он се након додељивања никада не мења.



Ако немате у виду колону или скуп колона које би могле да буду добар или природни примарни кључ, можете користити тип података „**AutoNumber**“, где ће систем аутоматски доделити вредност.

Пример на слици колоне 1 : тип података је постављен на „**AutoNumber**“, где се ниједан ID производа не понавља.

У неким случајевима, можда ћете желети да за примарни кључ табеле користите два или више поља. На пример, табела „**Detalji porudzbine**“ у којој су ускладиштене ставке за поруџбину би за примарни кључ користила две колоне: „**ID porudzbine**“ и „**ID proizvoda**“. Када се примарни кључ састоји из више колона, назива се и сложени или композитни кључ.

У базама података продаје производа можете да креирате колону „**AutoNumber**“ која ће у свакој табели служити као примарни кључ: „**ID proizvoda**“ за табелу „**Proizvodi**“, „**ID porudzbine**“ за табелу „**Narudzbine**“, „**ID kupca**“за табелу „**Kupc**i“ и „**ID dobavljaca**“ за табелу „**Dobavljaci**“.

6. Креирање релација између табела

Релације између табела се креирају уз помоћ примарног и страног кључа чиме се постиже интегрисани систем података. Након распоређивања информација у табеле, потребан вам је метод да их поново повежете на смислен начин.

На пример, следећи образац садржи информације из неколико табела:

1 Информације у овом обрасцу потичу из табеле „**Klijenti**“...

2 ...табеле „**Zaposleni**“...

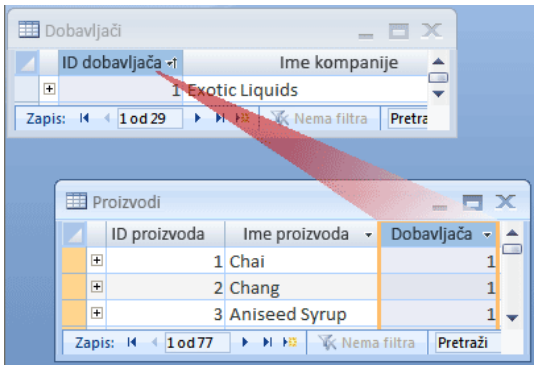
3 ...табеле „**Narudzbine**“...

4 ...табеле „**Proizvodi**“...

5 ...и табеле „**Detalji porudzbine**“.

У релационој бази података информације се на основу теме деле у засебне табеле. Након тога се користе релације између табела како би се информације по потреби повезале.

а) Креирање релације „један-према-више“



Размотрите овај пример: табеле „**Dobavljaci**“ и „**Proizvodi**“. Добављач може да доставља неограничен број производа. То значи да за сваког добављача који је представљен у табели „**Dobavljaci**“ може да постоји више производа представљаних у табели „**Proizvodi**“. Дакле, релација између табеле „**Dobavljaci**“ и табеле „**Proizvodi**“ је релација „**један-према-више**“.

Да бисте представили релацију „**један-према-више**“ у дизајну базе података, примарни кључ са стране релације „**један**“ додајте у виду једне или више додатних колона у табелу на страни релације „**више**“. У овом случају, нпр. колону „**ID dobavljaca**“ додајете у табелу „**Proizvodi**“. Систем ће након тога моћи да користи ID број добављача у табели „**Proizvodi**“ за проналажење одговарајућег добављача за сваки производ. Колона „**ID dobavljaca**“ у табели „**Proizvodi**“ назива се страни кључ.

б) Креирање релације „више-према-више“

Размотрите релацију између табеле „**Proizvodi**“ и табеле „**Narudzbine**“. Једна поруџбина може да садржи више производа, а један производ може да се појави у више поруџбина. Стога, за сваки запис у табели „**Narudzbine**“ може да постоји више записа у табели „**Proizvodi**“, док за сваки запис у табели „**Proizvodi**“ може да постоји више записа у табели „**Narudzbine**“.

Овај тип релације назива се релација „**више-према-више**“ јер за сваки производ може да постоји више поруџбина, а за сваку поруџбину може да постоји више производа. Имајте у виду да је за откривање релације „**више-према-више**“ између табела важно да узмете у обзир обе стране релације.

Теме табела поруџбина и производа имају тако релацију „**више-према-више**“.

То представља проблем. Како бисте схватили проблем, замислите шта би се догодило ако бисте покушали да креирате релацију између ове две табеле додавањем поља „**ID proizvoda**“ у табелу „**Narudzbine**“.

Потребно је да у табели „**Narudzbine**“ постоји више од једног записа по поруџбини да бисте имали више производа по поруџбини. Понављаћете информације о поруџбини за сваки ред који је повезан са појединачном поруџбином – што за последицу има неефикасан дизајн који би могао да доведе до нетачних података. Срешћете се са истим проблемом ако поље „**ID porudzbine**“ сместите у табелу „**Proizvodi**“ – за сваки производ ћете имати више од једног записа у табели „**Proizvodi**“. Како решити овакав проблем?

Одговор лежи у креирању треће табеле која се често назива везна табела, а која релацију „**више-према-више**“ рашчлањује на две релације „**један-према-више**“. Примарни кључ из обе табеле умеће се у трећу табелу. Као последица тога, трећа табела ће записати свако појављивање или инстанцу релације.

Сваки запис у табели „**Detalji porudzbine**“ представља једну ставку у поруџбини. Примарни кључ табеле „**Detalji porudzbine**“ састоји се из два поља – страних кључева из табеле „**Narudzbine**“ и табеле „**Proizvodi**“. Као примарни кључ ове табеле не можете користити само поље „**ID porudzbine**“, јер једна поруџбина може да има више ставки. Поље „**ID porudzbine**“ се понавља за сваку ставку у поруџбини, тако да оно не садржи јединствене вредности. Не можете користити ни само поље „**ID proizvoda**“, јер један производ може да се појави у више поруџбина. Али ова два поља заједно увек креирају јединствену вредност за сваки запис.

У бази података продаје производа, табела „**Narudzbine**“ и табела „**Proizvodi**“ нису директно повезане. Оне су, у ствари, повезане индиректно преко табеле „**Detalji porudzbine**“. Релација „**више-према-више**“ између поруџбина и производа представљена је у бази података помоћу две релације „**један-према-више**“:

- Између табела „**Narudzbine**“ и „**Detalji porudzbine**“ постоји релација „**један-према-више**“. Свака поруџбина може да има више ставки, док је свака ставка повезана са само једном поруџбином.
- Између табела „**Proizvodi**“ и „**Detalji porudzbine**“ постоји релација „**један-према-више**“. Сваки производ може да има више придружених ставки, али свака ставка упућује само на један производ.

Из табеле „**Detalji porudzbine**“ можете да одредите све производе у одређеној поруџбини. Можете да одредите и све поруџбине за одређени производ.

в) Креирање релације „један-према-један“

Релација „**један-према-један**“ се ретко користи у пракси. Али претпоставимо да треба да запишемо неке специјалне допунске информације о производу које ће нам ретко бити потребне или које су применљиве на само неколико производа. Пошто вам те информације нису често потребне и пошто би њихово складиштење у табелу „**Proizvodi**“ довело до појаве празног простора за сваки производ за који оне не важе, сместите их у одвојену табелу. Користите „**ID proizvoda**“ као примарни кључ, као у табели „**Proizvodi**“. Релација између ове допунске табеле и табеле „**Proizvodi**“ је релација „**један-према-један**“. За сваки запис у табели „ **Proizvodi**“ постоји један подударни запис у допунској табели. Када идентификујете такву релацију, обе табеле морају да деле заједничко поље.

Када откријете потребу за релацијом „**један-према-један**“, размотрите да ли информације из две табеле можете да сместите у једну табелу.

Ако то из неког разлога не желите да урадите, можда због тога што би то произвело много празног простора, следећа листа садржи опис начина на који бисте ту релацију представили у дизајну:

- Ако две табеле имају исту тему, вероватно можете да подесите релацију помоћу истог примарног кључа у обе табеле.
- Ако две табеле имају различите теме са различитим примарним кључевима, одаберите једну од табела (било коју) и уметните њен примарни кључ у другу табелу као страни кључ.

Да резимирамо, када постоји релација „**један-према-један**“ или „**један-према-више**“, биће потребно да табеле унутар релације деле заједничку колону или колоне. Када постоји релација „**више-према-више**“, за представљање релације ће бити потребна трећа табела.

7. Прочишћавање дизајна базе

После дизајна табела, треба поново простудирати структуру базе, и отклонити евентуалне недостаке и мане. Ово је прави тренутак за то, јер после уношења података, измена структуре базе је знатно отежана. Ако имате потребне табеле, поља и релације, требало би да креирате табеле и попуните их пробним подацима, као и да испробате рад са информацијама: креирање упита, додавање нових записа итд. На тај начин ћете допринети откривању потенцијалних проблема – на пример, можда ће бити потребно да додате колону коју сте заборавили да уметнете током фазе дизајнирања или можда би неку табелу требало да поделите на две табеле како бисте уклонили дуплирање.

Проверите да ли базу података можете да користите како бисте добили жељене одговоре. Креирајте прве радне верзије образаца и извештаја и видите да ли оне приказују очекиване резултате. Потражите непотребне дупликате података и измените дизајн како бисте их елиминисали након што их пронађете. Док будете испробавали почетну базу података, вероватно ћете открити да има места за побољшања.

Ево неколико ствари које би требало да проверите:

- Да ли сте заборавили неке колоне? Ако је то случај, да ли те информације припадају постојећим табелама? Уколико се ради о информацијама о нечем другом, можда ће бити потребно да креирате још једну табелу. Креирајте колону за сваку ставку информација коју треба да пратите. Ако се та информација не може израчунати на основу других колона, постоји вероватноћа да ће вам за њу бити потребна нова колона.
- Да ли су неке колоне непотребне јер се могу израчунати на основу постојећих поља? Ако нека ставка информација може да се израчуна на основу других постојећих колона – нпр. цена са попустом израчуната на основу малопродајне цене – често је боље урадити само то и избећи креирање нове колоне.
- Да ли изнова уносите дуплиране информације у неку табелу? Ако је то случај, вероватно је потребно да табелу поделите на две табеле између којих ће постојати релација „један-према-више“.
- Да ли имате табеле са великим бројем поља, ограниченим бројем записа и много празних поља у појединачним записима? Ако је то случај, размотрите поновно дизајнирање табеле у којој ће бити мање поља и више записа.
- Да ли сте сваку информацију рашчланили на најмање корисних делова? Ако је потребно да извештавате, сортирате, претражујете или израчунавате према некој ставци информације, ставите је у засебну колону.
- Да ли свака колона садржи чињеницу о теми табеле? Ако колона не садржи информацију о теми табеле, она припада другој табели.
- Да ли су све релације између табела представљене, било помоћу заједничких поља, било помоћу треће табеле? Релације „један-према-један“ или „један-према-више“ захтевају заједничке колоне. Релације „више-према-више“ захтевају трећу табелу.

Прочишћавање табеле „Proizvodi“:

Претпоставимо да сваки производ у бази података продаје производа спада у општу категорију, попут алкохолних и безалкохолних пића, зачина или морских плодова. Табела „**Proizvodi**“ би могла да садржи поље које приказује категорију сваког производа.
Претпоставимо да после испитивања и прочишћавања дизајна базе података одлучите да ускладиштите опис категорије са њеним именом. Ако додате поље „**Opis kategorije**“ у табелу „**Proizvodi**“, мораћете да понављате опис категорије за сваки производ који спада у одређену категорију – ово није добро решење.
Боље решење је да табела „**Kategorije**“ постане нова тема која ће се пратити у бази података, са сопственом табелом и сопственим примарним кључем. Након тога примарни кључ из табеле „**Kategorije**“ можете додати у табелу „**Proizvodi**“ као страни кључ.
Табеле „**Kategorije**“ и „**Proizvodi**“ имају релацију „**један-према-више**“: категорија може да садржи више производа, али производ може да припада само једној категорији.

8. Побољшање дизајна применом правила нормализације

Правила нормализације података (понекад се називају само правила нормализације) могу да се примене као следећи корак у дизајну да би се проверило да ли су табеле исправно структуриране. Процес примењивања правила на дизајн базе података назива се нормализација базе података или само нормализација. Нормализација је најкориснија пошто представите све ставке информација и дођете до прелиминарног дизајна. Циљ је да се уверите да сте информације поделили у одговарајуће табеле.

Постоји пет нивоа нормализације података:
1. **Елиминисање понављајућих група** (1НФ, или прва нормална форма)
2. **Елиминисање редундантних података** (2НФ, или друга нормална форма)
3. **Елиминисање колона које нису зависне од примарног кључа** (3НФ, или трећа нормална форма)
4. Изолација независних вишеструких релација (4НФ, или четврта нормална форма);
5. Изолација семантички повезаних вишеструких релација (5НФ, или пета нормална форма).

Правила примењујете редом, при чему се током сваког корака морате уверити да је ваш дизајн постигао такозвану „**нормалну форму**“. Општеприхваћено је пет таквих форми – од прве до пете нормалне форме, где су прве три углавном довољне за већину случајева дизајна базе података.

а) **Прва нормална форма**

Прва нормална форма подразумева да у сваком пресеку реда и колоне у табели постоји једна вредност, а никада листа вредности. На пример, не можете имати поље под називом „**Cena**“ у које ћете сместити више цена. Ако сваки пресек редова и колона замислите као ћелију, свака ћелија може да садржи једну вредност.

б) **Друга нормална форма**

Друга нормална форма захтева да свака колона без кључа потпуно зависи од целог примарног кључа, а не само неког његовог дела. Ово правило се примењује када имате примарни кључ који се састоји од више колона. На пример, претпоставимо да имате табелу која садржи следеће колоне, при чему поља „**ID porudzbine**“ и „**ID proizvoda**“ формирају примарни кључ:

- ID porudzbine (примарни кључ)
- ID proizvoda (примарни кључ)
- Ime proizvoda

Овакав дизајн крши правило друге нормалне форме, јер колона „**Ime proizvoda**“ зависи од поља „**ID proizvoda**“, а не од поља „**ID porudzbine**“, тако да не зависи од целог примарног кључа. Потребно је да из табеле уклоните колону „**Ime proizvoda**“. Она припада другој табели (табели „**Proizvodi**“).

в) **Трећа нормална форма**

Трећа нормална форма не захтева само да свака колона без кључа буде зависна од целог примарног кључа, већ и да колоне без кључа зависе једна од друге.

Другим речима, свака табела без кључа мора да зависи од примарног кључа и ни од чега другог. На пример, претпоставимо да имате табелу која садржи следеће колоне:

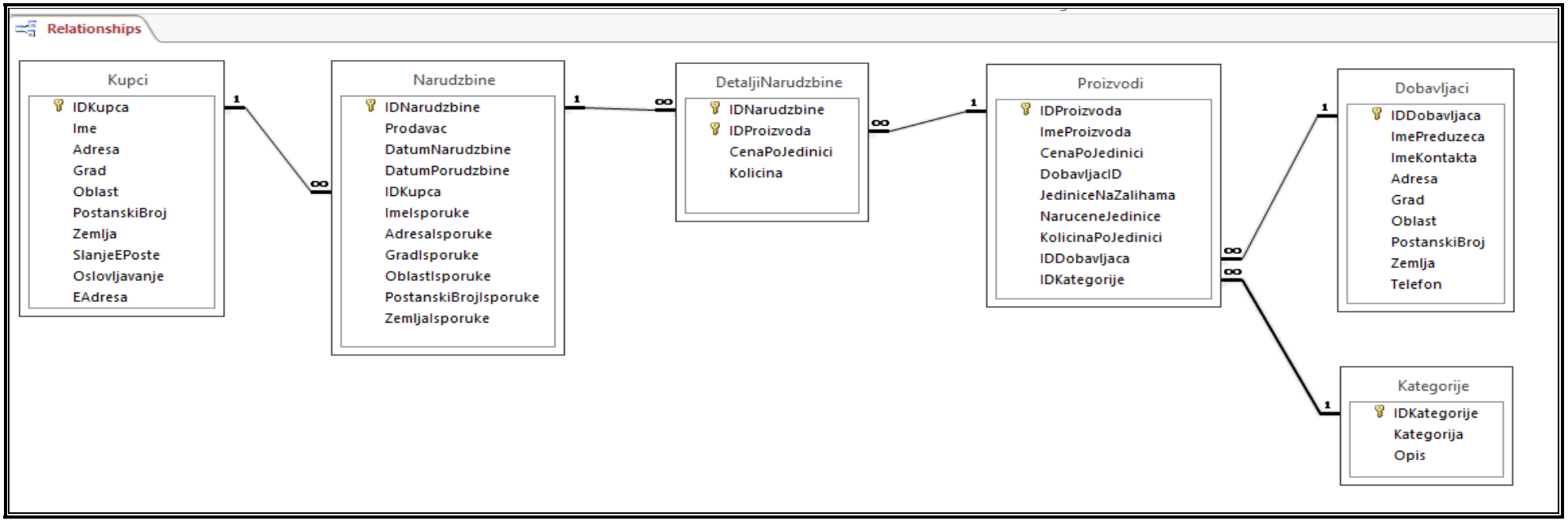
- ID proizvoda (примарни кључ)
- Ime
- PMC
- Popust

Претпоставимо да колона „**Popust**“ зависи од препоручене малопродајне цене (PMC). Ова табела нарушава трећу нормалну форму јер колона без кључа „**Popust**“ зависи од колоне без кључа „**PMC**“. Независност колоне значи да би требало да будете у могућности да промените сваку колону која не садржи кључ не утичући на друге колоне. Ако промените вредност у пољу „**PMC**“, колона „**Popust**“ ће се променити у складу са тим и на тај начин прекршити то правило. У овом случају, колону „**Popust**“ би требало преместити у другу табелу са кључем у пољу „PMC“.

9. Креирање осталих објеката базе података и уношење података

Ако је структура базе (табеле, поља, релације) задовољавајућа, приступа се уношењу података. За то се обично креирају обрасци (**forms**). Осим њих могу се креирати и остали објекти базе података и то: упити (**queries**), извештаји (**reports**), стране за приступ подацима (**data access pages**), макрои (**macros**) и модули (**modules**).

Коначан дизајн базе података везане за пословање предузећа „Северна звезда“.



III. Microsoft Access

1. Шта је то Access?

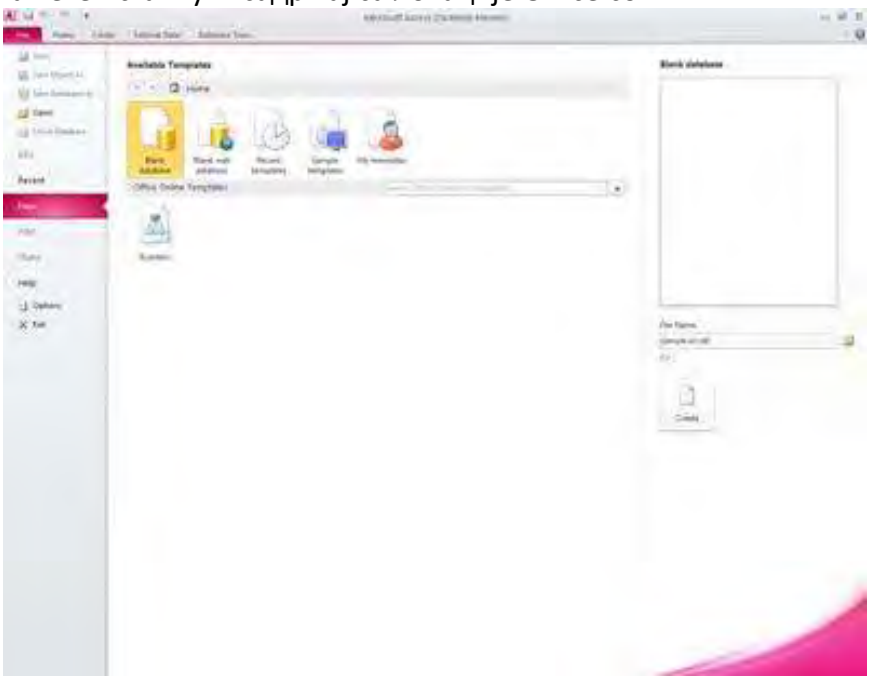
Microsoft Access је програм за дизајн и примену апликација за базе података коју можете да користите да бисте пратили важне информације. Податке можете да чувате на рачунару или да их објавите на Вебу – како би други могли да користе вашу базу података у Веб прегледачу.

Многе особе почињу да користе Access када програм који употребљавају за праћење неких ставки постаје све мање погодан за тај задатак. На пример, претпоставимо да се бавите планирањем догађаја и желите да пратите све детаље потребне за управљање догађајима како би они били успешни. Ако за то користите програм за обраду текста или програм за унакрсне табеле, лако можете наићи на проблеме са дуплираним и недоследним подацима. Можете користити софтвер за календаре, али праћење финансијских информација у календару није одговарајуће решење.

Microsoft Access подржава и рад са релационим базама података – где су подаци подељени на мање колекције података (називају се табеле) да би се избегла редундантност, а затим повезани на основу заједничких делова информација (називају се поља). На пример, релациона база података о планирању догађаја може да садржи табелу са информацијама о клијентима, табелу са информацијама о добављачима, као и табелу са информацијама о догађајима. Табела са информацијама о догађајима може имати поље које је повезано са табелом о клијентима и поље повезано са табелом о добављачима. На тај начин, ако се промени број телефона добављача, информације могу да се промене једном у табели о добављачима, а не у сваком догађају који укључује тог добављача.

Access је алатка коју можете користити за брз и лак развој апликација за релационе базе података које вам помажу у управљању информацијама. Можете креирати базу података која ће вам помоћи да пратите скоро све врсте информација, на пример залихе, професионалне контакте или пословне процесе. У ствари, Access садржи шаблоне које одмах можете користити за праћење разних информација, што је лако чак и за почетника.

Када покренете Access 2010, видећете приказ „**Microsoft Office Backstage**“ где можете да добијете информације о тренутној бази података, креирате нову базу података, отворите постојећу базу података или прикажете истакнути садржај са локације Office.com.



Приказ „**Backstage**“ садржи и многе друге команде које можете користити за прилагођавање, одржавање или дељење база података. Уопштено узевши, команде у приказу „**Backstage**“ примењују се на целе базе података, а не на појединачне објекте у бази података.

НАПОМЕНА У приказ „**Backstage**“ можете прећи било када тако што ћете изабрати картицу **File**.

1. Креирање базе података испочетка

Да бисте креирали нову базу података, урадите следеће:

1. Покрените Access.
2. У приказу „**Backstage**“, на картици **New** кликните на дугме **Blank Database**.
3. Са десне стране, у пољу **File Name** откуцајте име базе података.

Да бисте променили локацију на којој креирате фајл, кликните на дугме **Browse**  поред поља **File Name**, потражите и изаберите нову локацију, а затим кликните на дугме **OK**.

4. Кликните на дугме **Create**.

Access креира базу података, а затим отвара празну табелу (под именом „Table1“) у приказ листа са подацима.

5. Access поставља курсор у прву празну ћелију у колони. Да бисте додали податке, почните да куцате.

НАПОМЕНА Уношење података у приказу листа са подацима дизајнирано је тако да буде слично уношењу података у Excel радни лист. Главно ограничење је то што подаци морају да се унесу у суседне редове и колоне, почевши од горњег левог угла листа са подацима. Не би требало да покушавате да обликујете податке додавањем празних редова или колона, као што бисте то можда урадили у Excel радном листу, јер ћете на тај начин простор у табели остати неискоришћен. Табела садржи само податке. Свако визуелно представљање тих података биће реализовано у обрасцима и извештајима које ћете дизајнирати касније.

Структура табеле се креира док уносите податке. Сваки пут када у лист са подацима додате нову колону, дефинише се ново поље. Access поставља тип податка поља на основу типа података које унесете. На пример, ако сте у једну колону уносили само вредности датума, Access ће за то поље поставити тип података „Date/Time“. Ако у њега касније покушате да унесете вредност која није датум (као што је име или број телефона), Access приказује поруку са обавештењем да се вредност не подудара са типом података те колоне. Ако је могуће, требало би да планирате табелу тако да свака колона садржи исти тип података, без обзира на то да ли се ради о тексту, датумима, бројевима или неком другом типу. То знатно олакшава прављење упита, образаца и извештаја који бирају само жељене податке.

Ако сте завршили са унесом података, кликните на дугме **Close**  .

НАПОМЕНА Access ће избрисати табелу „Table1“ ако је затворите без чувања.

Отварање постојеће Access базе података

САВЕТ Да бисте брзо отворили неку од неколико база података које сте недавно отварали, на картици **File** кликните на дугме **Recent**, а затим изаберите име фајла.

1. На картици **File** кликните на дугме **Open**.
2. Кликните на пречицу у дијалогу **Open** – или са листе **Browse** изаберите диск јединицу или фолдер која садржи жељену базу података.
3. На листи фолдера кликћите двапут на фасцикле док не отворите онај фолдер која садржи жељену базу података.
4. Када пронађете базу података, поступите на неки од следећих начина:
   - Кликните двапут на базу података да бисте је отворили у подразумеваном режиму отварања.
   - Да бисте базу података отворили за дељени приступ у база података за више корисника (дељена) окружењу, тако да и ви и други корисници у бази података можете истовремено да читате и пишете, кликните на дугме **Open**.
   - Кликните на стрелицу на дугмету **Open**, а затим изаберите ставку **Open Read-Only** да бисте базу података отворили са приступом само за читање, тако да можете да је прегледате, али не и да је уређујете.
   - Кликните на стрелицу на дугмету **Open**, а затим изаберите ставку **Open Exclusive** да бисте базу података отворили са искључивим приступом, тако да друге особе не могу да је отворе док је отворена код вас.
   - Кликните на стрелицу на дугмету **Open**, а затим изаберите ставку **Open Exclusive Read-Only** да бисте базу података отворили са приступом само за читање. Други корисници ће и даље моћи да отворе базу података, али ће бити ограничени на приступ само за читање.

Ако не можете да пронађете базу података коју желите да отворите, онда:

1. У дијалогу **Open** кликните на пречицу **My Computer (My PC)** – или са листе **Browse** изаберите ставку **My Computer**.
2. На листи диск јединица кликните десним тастером миша на ону диск јединицу за коју верујете да садржи базу података, а затим кликните на дугме **Search**.
3. Унесите критеријуме претраге, а затим притисните тастер ENTER да бисте потражили базу података.
4. Ако је база података пронађена, кликните двапут на њу у дијалогу **Search Results** да бисте је отворили.

IV. Табеле

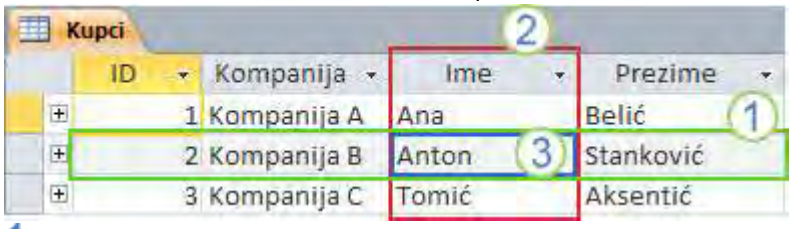
1. Увод у табеле

Када користите базу података, податке складиштите у табелама – листама заснованим на темама које садрже податке распоређене у записима. На пример, можете креирати табелу „**Контакти**“ за складиштење листе имена, е-адреса и бројева телефона.

Дизајнирање базе података би требало да почнете тако што ћете испланирати све њене табеле и одлучити на који начин ће оне међусобно бити повезане. Пре креирања табела, пажљиво размотрите своје захтеве и утврдите које су вам табеле потребне.

Табела је објекат базе података који користите за складиштење података о одређеној теми, на пример о запосленима или производима. Табела се састоји од записа и поља. Сваки запис садржи податке о једној инстанци теме табеле, на пример о одређеном запосленом. Запис се такође често назива ред или инстанца. Свако поље садржи податке о једном аспекту теме табеле, на пример о имену или е-адреси. Поље се такође често назива колона или атрибут.

Запис се састоји од вредности поља, на пример Contoso, Ltd. или neko@example.com. Вредност поља се обично назива и чињеница.



1 Запис
2 Поље
3 Вредност поља

База података може да садржи много табела од којих свака складишти информације о другој теми. Свака табела може да садржи велики број поља са различитим типовима података, као што су текст, бројеви, датуми и хипервезе.

2. Креирање и рад са табелама

Креирајте нову табелу ако имате нови извор података који не припада ниједној од постојећих табела. Табелу можете креирати тако што ћете креирати нову базу података, уметнути табелу у постојећу базу података, увести табелу из другог извора података или се повезати са њом – као што је Microsoft Office Excel радна свеска, Microsoft Office Word документ, текстуална датотека, Веб услуга или нека друга база података. Када креирате нову, празну базу података, у њу се аутоматски умеће нова, празна табела. Након тога можете унети податке у табелу да бисте почели да дефинишете поља.

а) Креирање нове табеле у новој бази података

1. На картици **File** изаберите ставку **New**, а затим кликните на дугме **Blank Database**.
2. У пољу **File name** откуцајте име фајла за нову базу података.
3. Да бисте потражили другу локацију за чување базе података, кликните на икону фасцикле.
4. Кликните на дугме **Create**.

Отвара се нова база података и креира се нова табела под именом „Table1“ која се отвара у приказу листа са подацима.

б) Креирање нове табеле у постојећој бази података

1. Изаберите картицу **File**, поставите показивач на ставку **Open** и урадите нешто од следећег:

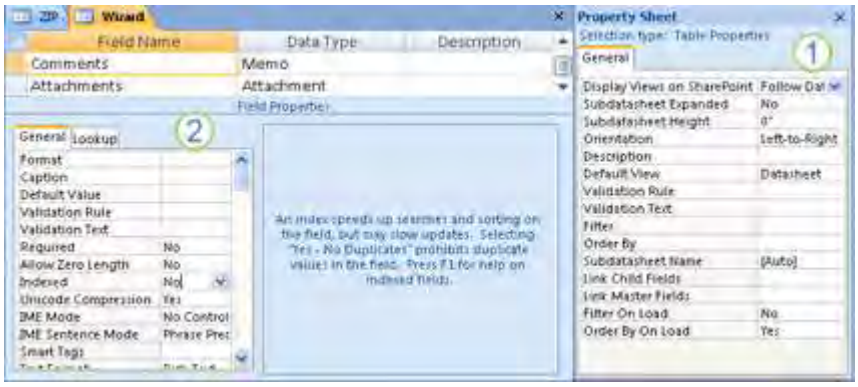
Ако је жељена база података наведена у оквиру **Recent Databases** (**Недавне базе података)**, са те листе изаберите базу података, -или- ако база података није наведена у оквиру **Recent Databases**, у оквиру **Open Document** изаберите жељену опцију.

2. У дијалогу **Open** изаберите базу података коју желите да отворите и кликните на дугме **Open**
3. На картици **Create**, у групи **Tables** кликните на дугме **Table**.

Нова табела се умеће у базу података и отвара у табеларном приказу.

3. Својства табеле и поља

Табеле и поља имају својства која можете подесити да контролишу њихове карактеристике или понашање.



Табела отворена у приказу дизајна.

1 Својства табеле

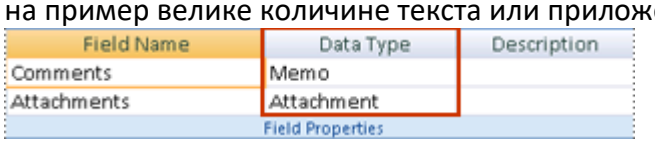
2 Својства поља

У Access бази података, својства табеле су атрибути табеле који утичу на изглед или понашање табеле као целине. Својства табеле су подешена у листу са својствима табеле, у приказу дизајна. На пример, својство табеле **Default View** можете подесити да наводи подразумевани приказ табеле.

Својство поља се примењује на одређено поље у табели и дефинише неку од карактеристика поља или аспект понашања поља. Нека својства поља можете подесити у приказу листа са подацима. Било које својство поља такође можете подесити у приказу дизајна помоћу панела **Field Properties**.

4. Типови података

Свако поље има тип података. Тип података поља указује на врсту података које поље складишти, на пример велике количине текста или приложених датотека.



Тип података је својство поља, али разликује се од других својстава поља на следећи начин:

- Тип података поља постављате у координатној мрежи дизајна табеле, а не у окну **Field Properties**.
- Тип података поља утврђује друга својства која поље има.
- Када креирате поље, морате поставити тип података поља.

НАПОМЕНА Ново поље можете креирати у програму Access тако што ћете податке унети у нову колону у табеларном приказу са подацима. Када поље креирате уносом података у табеларном приказу, Access пољу аутоматски додељује тип података заснован на вредности коју унесете. Ако унос не подразумева ниједан други тип података, Access тип података поставља на вредност „Text“.

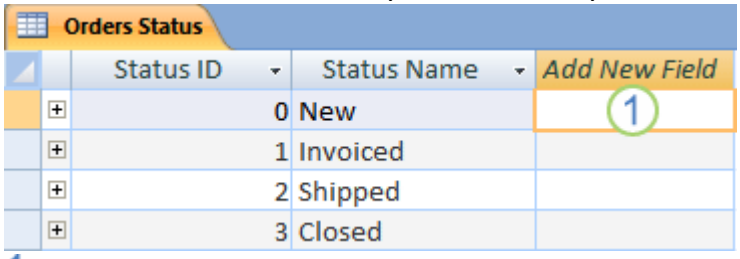
5. Додавање поља у табелу

Сваки део података који желите да пратите, складиштите у пољу. На пример, у табели контаката креирате поља за презиме, име, број телефона и адресу. У табели производа креирате поља за име производа, ID proizvoda и цену.

Пре креирања поља, покушајте да рашчланите податке на најмање корисне делове. Касније је много једноставније комбиновати податке, него их рашчлањивати. На пример, уместо поља за пуно име, размотрите креирање засебних поља за презиме и име. У том случају можете једноставно вршити претрагу или сортирање по имену, презимену или и једном и другом. Ако планирате да извештавате о некој ставци података, да је сортирате, претражујете или израчунавате, у поље поставите само ту ставку. Када креирате поље, можете да поставите и својства поља да бисте контролисали његов изглед и понашање.

Додавање поља уносом података

Када креирате нову табелу или отворите постојећу табелу у табеларном приказу, у табелу можете додати поље тако што ћете унети податке у колони **Add New Field**.



1 Унесите податке у колону **Add New Field**.

1. Креирајте табелу или је отворите у приказу листа са подацима.
2. У колони **Add New Field** унесите име поља које желите да креирате. Користите описно име да би се поље лакше идентификовало.
3. Унесите податке у ново поље.

6. Постављање својстава поља

Када креирате поље, можете да поставите својства поља да бисте контролисали његов изглед и понашање.

На пример, постављањем својстава поља можете:

- Контролисати изглед података у пољу
- Допринети спречавању уноса неисправних података у поље
- Навести подразумеване вредности поља
- Допринети убрзавању претраживања и сортирања поља

Нека доступна својства поља можете да поставите док радите у приказу листа са подацима. Међутим, да бисте имали приступ и да бисте поставили комплетну листу својстава поља, морате да користите приказ дизајна.

Постављање својстава поља у табеларном приказу

Током рада у приказу листа са подацима можете преименовати поље, променити му тип података, променити му својство **Format**, као и нека друга својства поља.

Отварање табеле у приказу листа са подацима

1. У окну за навигацију кликните десним тастером миша на табелу коју желите да отворите.
2. У приручном менију изаберите ставку **Datasheet view**.

Преименовање поља

Када поље додате уносом података у приказу листа са подацима, Access пољу аутоматски додељује генеричко име. Access првом новом пољу додељује име „**Field1**“, другом новом пољу име „**Field2**“ и тако даље. Име поља се подразумевано користи као његова ознака где год да се поље приказује,

на пример, у наслову колоне на листу са подацима. Ако преименујете поља и дате им описнија имена, лакше ћете их користити приликом приказивања или уређивања записа.

1. Кликните десним тастером миша на наслов поља које желите да преименујете (на пример, „**Field1**“).
2. У приручном менију изаберите ставку **Rename Column**.
3. У наслов поља унесите ново име.

Имена поља се могу састојати од највише 64 знака (слова или бројева), укључујући размаке.

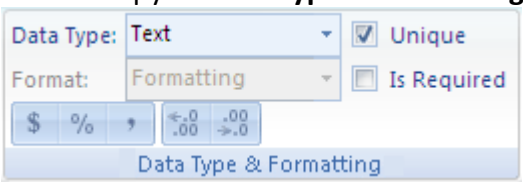
Промена типа података поља

Када поље креирате уносом података у приказ листа са подацима, Access испитује те податке како би утврдио одговарајући тип података за то поље. На пример, ако унесете **1/1/2006**, Access те податке препознаје као датум и поставља тип података поља на вредност „Date/Time“. Ако Access не може поуздано да утврди тип података, тип података се подразумевано поставља на вредност „Text“.

Тип података поља утврђује остала својства поља која можете поставити. На пример, за поље које има тип података „Hiperlink“ или „Мемо“ можете поставити само својство **Append Only**.

У неким случајевима ћете можда желети ручно да промените тип података поља. На пример, претпоставимо да имате бројеве соба који изгледају као датуми, на пример 10/2001. Ако у ново поље у приказу листа са подацима унесете**10/2001**, функција аутоматског откривања типа података за то поље бира тип података „Датум/време“. Пошто су бројеви соба ознаке, а не датуми, требало би да користе тип података „Text“. Користите следећу процедуру да бисте променили тип података поља.

1. Изаберите картицу **Fields**.
2. У групи **Data Type & Formatting**, са листе **Data Type** изаберите жељени тип података.



Доступни типови података

Тип података	Користи се за складиштење	Величина
Text (Текст)	Алфанумерички знакови. Користе се за текст или за бројеве који се не користе у израчунавањима (на пример, ID производа). Нумеричка вредност која је ускладиштена као текст може да се сортира и филтрира логичније, али не може лако да се користи у израчунавањима.	До 255 знакова.
Memo (Меморандум)	Алфанумерички знакови (дужине веће од 255 знакова) или текст који користи обликовање обогаћеног текста. Користе се за текст који је дужи од 255 знакова или за текст који користи обликовање обогаћеног текста. Белешке, дуги описи и пасуси који користе обликовање текста, на пример подебљани текст или курзив, добри су примери употребе поља „**Memo**“.	До 1 гигабајт знакова или 2 гигабајта простора за складиштење (2 бајта по знаку), од којих можете приказати 65.535 знакова у контроли.
Number (Број)	Нумеричке вредности: Цели: Byte, Integer, Long Integer или	1, 2, 4 или 8.

Тип података	Користи се за складиштење	Величина
	Децимални бројеви: Single, Double, Decimal.	
Date/Time (Датум/време)	Датуми и времена. Користе се за складиштење вредности датума/времена. Обратите пажњу на то да свака ускладиштена вредност обухвата компоненту датума и компоненту времена.	8 бајтова.
Currency (Валута)	Новчане вредности. Користи се за складиштење новчаних вредности (валута).	8 бајтова.
AutoNumber (Аутоматски број)	Јединствена нумеричка вредност коју Access аутоматски умеће када се дода запис. Користи се за генерисање јединствених вредности које се могу користити као примарни кључ. Обратите пажњу на то да вредности за поља аутоматског броја могу да се увећавају постепено или за наведену вредност или да се насумично доделе.	4 бајта.
Yes/No (Да/Не)	Булове вредности (да/не). Можете користити један од три облика: Yes/No (Да/Не), True/False (Тачно/Нетачно) или On/Off (Укључено/Искључено).	1 бит (8 битова = 1 бајт).
Attachment (Прилог)	Слике, бинарне фајлови, Office фајлови. Ово је пожељан тип података за складиштење дигиталних слика и било ког типа бинарних датотека.	За компримоване прилоге 2 гигабајта. За некомпримоване прилоге око 700 кБ, у зависности од степена до којег се прилог може компримовати.
Hyperlink (Хипервеза)	Хипервезе. Користе се за складиштење хипервеза да би се обезбедио приступ путем једног клика Веб страницама путем URL (Uniform Resource Locator) адресе или датотекама путем имена у UNC (universal naming convention) формату. Такође можете навести везе ка Access објектима ускладиштеним у бази података.	До 1 гигабајт знакова или 2 гигабајта простора за складиштење (2 бајта по знаку), од којих можете приказати 65.535 знакова у контроли.
Lookup Wizard (Чаробњак за проналажење)	Ово заправо није тип података, већ покреће чаробњак за проналажење. Користи се за покретање чаробњака за проналажење тако да можете да креирате поље које користи комбиновани оквир за тражење вредности у другој табели, упиту или листи вредности.	Засновано на табели или упиту: величина повезане колоне. Засновано на вредности: величина поља за текст које се користи за складиштење вредности.

НАПОМЕНА Максимална величина Access фајла базе података износи 2 гигабајта.

Савети о типовима података

- За бројеве телефона, бројеве делова и друге бројеве које не намеравате да користите за математичка израчунавања, уместо типа података „**Number**“ требало би да изаберете тип података „**Text**“. Нумеричка вредност ускладиштена као текст може се логичније сортирати и филтрирати, али је није могуће једноставно користити у израчунавањима.
- За типове података „**Text**“ и „**Number**“, величину поља или тип података можете прецизније навести тако што ћете поставити вредност у пољу за својство **FieldSize**.

Промена формата поља

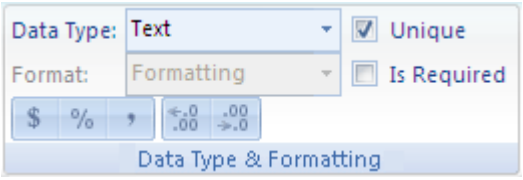
Поред утврђивања типа података новог поља, Access такође може поставити својство **Format** за поље, у зависности од тога шта унесете. На пример, ако унесете вредност „10:50“, Access поставља тип података на вредност „**Date/Time**“, а својство **Format** на вредност „**Medium Time**“. Да бисте ручно променили својство поља **Format**, урадите следеће:

1. На траци изаберите картицу **Fields**.
2. У групи **Data Type & Formatting**, на листу **Format** унесите жељени облик.

НАПОМЕНА Листа **Format** је можда недоступна за нека поља (на пример, текстуално поље), у зависности од типа података поља.

Постављање других својстава поља

1. У табеларном приказу изаберите поље за које желите да поставите својство.
2. На картици **Fields**, у групи **Data Type & Formatting** изаберите жељена својства.



Постављање својстава поља у приказу дизајна

Током рада са табелом у приказу дизајна можете поставити било које својство поља. У приказу дизајна тип података поља постављате у координатној мрежи дизајна табеле, а остала својства у окну **Fields properties**.

Отварање табеле у приказу дизајна

1. У окну за навигацију кликните десним тастером миша на табелу.
2. У приручном менију изаберите ставку **Design View**.

Промена типа података поља

1. У координатној мрежи дизајна табеле пронађите поље за које желите да поставите тип података.
2. У колони **Data Type**, са листе одаберите тип података.

Постављање других својстава поља

1. У координатној мрежи дизајна табеле изаберите поље за које желите да поставите својства. Access приказује својства овог поља у окну **Fields properties**.

НАПОМЕНА Тип података поља утврђује својства која можете поставити.

2. У окно **Fields properties** унесите жељене поставке за свако својство или притисните тастер F6 и користите тастере са стрелицама да бисте изабрали својство.
3. Да бисте обезбедили више простора за унос или уређивање поставке својства у пољу за својство, притисните комбинацију тастера SHIFT+F2 да бисте приказали поље за **зумирање**.

Доступна својства поља

НАПОМЕНА Нису сва својства доступна за свако поље. Тип података поља одређује својства која то поље има.

Својство поља	Опис
Field Size (Величина поља)	Поставите максималну величину за податке ускладиштене у виду типа података „Text“, „Number“ или „Аутоматско нумерисање“ САВЕТ За најбоље перформансе увек наведите најмању довољну вредност за својство **Field Size**.
Format (Формат)	Прилагодите подразумевани начин појављивања поља када се прикаже или одштампа.
Decimal Places (Децимална места)	Наведите број децималних места која се користе приликом приказивања бројева.
New Values (Нове вредности)	Наведите да ли се приликом додавања новог записа поље „**AutoNumber**“ повећава или му се додељује насумична вредност.
Input Mask (Маска за унос)	Прикажите знакове који помажу при уносу података.
Caption (Натпис)	Поставите текст који се подразумевано приказује у ознакама за обрасце, извештаје и упите.
Default Value (Подразумевана вредност)	Пољу аутоматски доделите подразумевану вредност приликом додавања нових записа.
Validation Rule (Правило за валидацију)	Унесите израз који мора бити тачан сваки пут када додате или промените вредност у овом пољу. Примери: >0, >=0, <0, <=0 – веће, веће једнако, мање, мање једнако нули респективно. >=100 And <500 – веће једнако од 100 И мање од 500 <100 Or >=500 – мање од 100 ИЛИ веће једнако од 500 (дијаметрално супротно од претходног правила.
Validation Text (Текст за валидацију)	Унесите поруку која се приказује када вредност крши израз у својству **Validation Rule**.
Required (Обавезно)	Захтевајте да подаци буду унети у поље.
Allow Zero Length (Дозвољавање нулте дужине)	Дозволите унос (постављањем вредности својства на вредност **Да**) низа нулте дужине (**""**) у текстуално или мемо поље.
Indexed (Индексирано)	Убрзајте приступ подацима у овом пољу креирањем и коришћењем индекса.

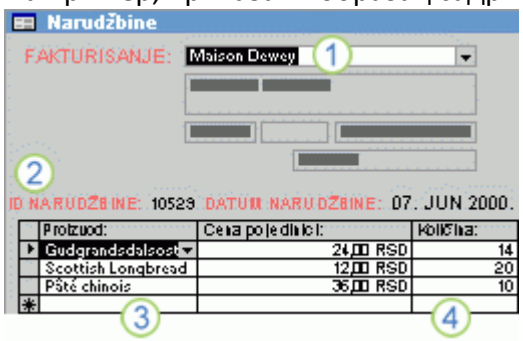
6. Водич за релације међу табелама

Један од циљева доброг дизајна базе података јесте уклањање редундантности података (дуплираних података). За постизање тог циља, потребно је да податке поделите у више табела заснованих на теми како би свака чињеница била представљена само једном. Затим програму Access обезбедите средства помоћу којих ће подељене информације поново спојити – то радите

постављањем заједничких поља у табеле које су повезане. Међутим, да бисте исправно извршили овај корак, прво морате разумети релације између табела, а затим навести те релације у бази података.

а) Увод

Када креирате табелу за сваку тему у бази података, програму Access морате обезбедити средства помоћу којих ће по потреби поново спојити те информације. Ово постижете постављањем заједничких поља у табеле које су повезане и дефинисањем релација између табела. Затим можете креирати упите, обрасце и извештаје који истовремено приказују информације из неколико табела. На пример, приказани образац садржи информације извучене из неколико табела:



1. Информације у овом обрасцу потичу из табеле „**Klijenti**“…
2. …табеле „**Narudzbine**“…
3. …табеле „**Proizvodi**“…
4. …и табеле „Detalji narudzbine“.

Име клијента у пољу **Испостави рачун** преузето је из табеле „**Klijenti**“, вредности за ID поруџбине и датум поруџбине преузете су из табеле „**Narudzbine**“, име производа потиче из табеле „**Proizvodi**“, а вредности за цену по јединици и количину потичу из табеле „**Detalji narudzbine**“. Ове табеле су међусобно повезане на више начина како би се информације из сваке од њих нашле у обрасцу.

У претходном примеру, поља у табелама морају бити координисана тако да приказују информације о истој поруџбини. Ова координација је постигнута коришћењем релација међу табелама. Релација међу табелама функционише тако што подудара податке у кључним пољима – често се ради о пољу са истим именом у обе табеле. У већини случајева, поља која се подударају су **примарни кључ** из једне табеле, који обезбеђује јединствени идентификатор сваког записа и **страни кључ** у другој табели. На пример, запослени могу бити повезани са поруџбинама за које су одговорни креирањем релације међу табелама између поља „**ID zaposlenog**“ у табелама „**Zaposleni**“ и „**Narudzbine**“.

1. Поље „**ID zaposlenog**“ појављује се у обе табеле – као примарни кључ ...
2. ... и као страни кључ.

б) Типови релација међу табелама

Постоје три типа релација међу табелама.

- **Релација један-према-више**

Размотрите базу података за праћење поруџбина која садржи табеле „**Klijenti**“ и „**Narudzbine**“. Клијент може да направи неограничен број поруџбина. Из тога следи да за сваког купца из табеле „**Klijenti**“ може да постоји више поруџбина у табели „**Narudzbine**“. Према томе, релација између табела „**Klijenti**“ и „**Narudzbine**“ представља релацију један-према-више.

Узмите примарни кључ на страни релације „један“ и додајте га као додатна поља у табелу на страни релације „више“ како бисте у дизајну базе података представили релацију један-према-више. У

овом случају, на пример, додајете ново поље – поље ID-а из табеле „**Klijenti**“ – у табелу „**Narudzbine**“ и дајете му име „**ID klijenta**“. Access затим може да користи број ID-ова клијената у табели „**Narudzbine**“ да би пронашао одговарајућег клијента за сваку поруџбину.

- **Релација више-према-више**

Размотрите релацију између табела „**Proizvodi**“ и „**Narudzbine**“. Једна поруџбина може садржати више производа. С друге стране, један производ може да се појави у више поруџбина. Према томе, за сваки запис у табели „**Narudzbine**“ може да постоји више записа у табели „**Proizvodi**“. Осим тога, за сваки запис у табели „**Proizvodi**“ може да постоји више записа у табели „**Narudzbine**“. Овај тип релације се назива релација више-према-више зато што за сваки производ може да постоји више поруџбина и за сваку поруџбину може постојати више производа. Имајте у виду да је важно да размотрите обе стране релације како бисте открили постојеће релације више-према-више између табела.

Мораћете да креирате трећу табелу, која се често назива везна табела, како бисте представили релацију више-према-више. Она дели релацију више-према-више на две релације један-према-више. Примарни кључ из обе табеле умећете у трећу табелу. Као резултат тога, трећа табела записује свако појављивање или инстанцу релације. На пример, табеле „**Narudzbine**“ и „**Proizvodi**“ имају релацију више-према-више која је дефинисана креирањем две релације један-према-више са табелом „**Detalji narudzbine**“. Једна поруџбина може имати више производа и сваки производ се може појавити на више поруџбина.

- **Релација један-према-један**

У релацији један-према-један сваки запис у првој табели може имати само један подударни запис у другој табели и сваки запис из друге табеле може имати само један подударни запис из прве табеле. Овај тип релације није уобичајен зато што су информације повезане на овај начин најчешће ускладиштене у истој табели. Релацију један-према-један можете користити да бисте поделили табелу са више поља, да бисте изоловали део табеле из безбедносних разлога или да бисте ускладиштили информације које се примењују само на подскуп главне табеле. Када идентификујете такву релацију, обе табеле морају да деле заједничко поље.

в) Зашто креирати релације међу табелама?

Релације међу табелама можете да креирате изричитим коришћењем прозора „**Relationships**“ или превлачењем поља из панела **Field List**. Access користи релације међу табелама да би знао како се спајају табеле када треба да их користите у објекту базе података. Постоји више разлога због којих треба да креирате релације међу табелама пре него што креирате друге објекте базе података, као што су обрасци, упити и извештаји.

- **Релације међу табелама утичу на дизајн упита**

Морате да креирате упит који спаја табеле да бисте радили са записима из више табела. Упит функционише тако што повезује вредности поља примарног кључа прве табеле са пољем споредног кључа у другој табели. На пример, да бисте добили редове који приказују све поруџбине за сваког купца, креираћете упит који на основу поља „**ID kupca**“ спаја табелу „**Klijenti**“ са табелом „**Narudzbine**“. У прозору „**Relationships**“ можете ручно да наведете поља која ће бити спојена. Осим тога, ако користите неки чаробњак за упите, Access употребљава информације које прикупља из релација међу табелама које сте већ дефинисали како би вам представио довољну количину информација на основу којих можете бирати, као и да би унапред попунио поставке својстава одговарајућим подразумеваним вредностима.

- **Релације међу табелама обезбеђују информације за дизајн образаца и извештаја**

Када дизајнирате образац или извештај, Access употребљава информације које прикупља из релација међу табелама које сте већ дефинисали како би вам представио довољну количину информација на основу којих можете бирати, као и да би унапред попунио поставке својстава одговарајућим подразумеваним вредностима.

- **Релације међу табелама представљају основу на коју можете наметнути референцијални интегритет да бисте у бази података спречили појаву записа који су сирочићи.**

Запис који је сироче јесте запис са референцом на други запис који не постоји – на пример, запис поруџбине који референцира запис клијента који не постоји.

Када дизајнирате базу података, информације делите у табеле од којих свака има примарни кључ. Након тога, у повезане табеле додајете споредне кључеве који референцирају ове примарне кључеве. Парови страни кључ-примарни кључ формирају основу релација међу табелама и упита за више табела. Према томе, важно је да референце страни кључ-примарни кључ буду синхронизоване. Референцијални интегритет помаже у одржавању синхронизованости референци и зависи од релација међу табелама.

г) Разумевање референцијалног интегритета

Када дизајнирате базу података, информације делите у више табела на основу теме како бисте умањили редундантност података. Затим програму Access обезбедите средства помоћу којих ће подаци поново спојити постављањем заједничких поља у повезане табеле. На пример, да бисте представили релацију један-према-више, узећете примарни кључ из табеле „један“ и додати га као додатно поље у табелу „више“. Да би повезао податке, Access преузима вредност из табеле „више“ и тражи одговарајућу вредност у табели „један“. На овај начин, вредности у табели „више“ референцирају одговарајуће вредности у табели „један“.

Претпоставимо да имате релацију „један-према-више“ између табела „Spediteri“ и „Narudzbine“ и желите да избришете неког шпедитера. Ако шпедитер кога желите да избришете има поруџбине у табели „Narudzbine“, оне ће постати „сирочићи“ када избришете запис. Narudzbine ће још увек садржати ID шпедитера, али ID више неће бити важећи зато што запис који он референцира више не постоји.

Сврха референцијалног интегритета јесте спречавање појаве сирочића и одржавање синхронизованости референци како до ове хипотетичке ситуације никада не би дошло.

Референцијални интегритет можете наметнути када га омогућите за релације међу табелама. Када је референцијални интегритет једном наметнут, Access одбија сваку операцију која би нарушила референцијални интегритет те релације међу табелама. То значи да ће Access одбити да изврши ажурирања која мењају одредиште референце, као и брисања која уклањају одредиште референце. Могуће је, међутим, да имате потпуно оправдану потребу да промените примарни кључ за шпедитера који има поруџбине у табели „Narudzbine“. У таквим случајевима потребно је да Access, као део једне операције, аутоматски ажурира све редове на које промена утиче. На тај начин Access обезбеђује извршавање комплетног ажурирања и конзистентност базе података, при чему су сви редови ажурирани. Из овог разлога, Access подржава опцију „**Cascade Update Related Fields**“. Када наметнете референцијални интегритет и одаберете опцију „**Cascade Update Related Fields**“, а затим ажурирате примарни кључ, Access ће аутоматски ажурирати сва поља која референцирају тај примарни кључ.

Такође је могуће да имате потпуно оправдану потребу да избришете ред и све повезане записе – на пример, запис шпедитера и све повезане поруџбине за тог шпедитера. Из тог разлога Access подржава опцију „**Cascade Delete Related Fields**“. Када наметнете референцијални интегритет и одаберете опцију „**Cascade Delete Related Fields**“, а затим избришете запис на страни примарног кључа релације, Access ће аутоматски избрисати све записе који референцирају тај примарни кључ.

д) Приказ релација међу табелама

На картици **Алатке базе података** кликните на дугме **Relationships** да бисте приказали релације међу табелама. Отвориће се прозор „Relationships“ у коме су приказане све постојеће релације. Ако још ниједна релација међу табелама није дефинисана, а прозор „Relationships“ отварате први пут, Access ће затражити од вас да у прозор додате табелу или упит.

Отварање прозора „**Relationships**“

1. На картици **File** кликните на дугме **Open**.

2. У дијалогу **Open** изаберите и отворите базу података.
3. На картици **Database Tools**, у групи **Relationships** кликните на дугме **Relationships**.
4. Ако база података садржи релације, појавиће се прозор „**Relationships**“. Ако база података не садржи ниједну релацију, а прозор „**Relationships**“ отварате први пут, појавиће се дијалог **Show Table**. Кликните на дугме **Close** да бисте затворили дијалог.
1. На картици **Design**, у групи **Односи** изаберите ставку **All Relationships**.

У прозору су приказане све дефинисане релације у бази података. Имајте у виду да скривене табеле (табеле за које је у дијалогу **Properties** потврђен избор у пољу за потврду **Hidden**) и њихове релације неће бити приказане, осим ако је у дијалогу **Navigation Options** потврђен избор у пољу за потврду **Show Hidden Objects**.

Релација међу табелама је представљена релационом линијом повученом између табела у прозору „**Relationships**“. Релација за коју није наметнут референцијални интегритет појављује се као танка линија између заједничких поља која подржавају ту релацију. Када изаберете ову релацију кликом на линију, линија ће постати дебља, што означава да је изабрана. Ако наметнете референцијални интегритет за ову релацију, линија ће постати дебља на крајевима. Осим тога, број **1** ће се појавити изнад дебљег дела линије на једној страни релације, а симбол за бесконачно (**∞**) изнад дебљег дела линије на другој страни.

Када је прозор „**Relationships**“ активан, на траци можете изабрати неку од следећих команди:

На картици **Design**, у групи **Tools**:

- **Edit Relationships** Отвара дијалог **Edit Relationships**. Када изаберете релациону линију, можете кликнути на дугме **Edit Relationships** да бисте променили релацију међу табелама. Можете такође двапут да кликнете на релациону линију.
- **Clear Layout** Уклања све табеле и релације из приказа у прозору **„Relationships“.** Имајте у виду да ова команда само сакрива табеле и релације – не брише их.
- **Relationships Report** Креира извештај који приказује табеле и релације у бази података. Извештај приказује само табеле и релације које нису скривене у прозору „**Relationships**“.

На картици **Design**, у групи **Relationships**:

- **Show Table** Отвара дијалог **Show Table** како бисте могли да изаберете табеле и упите за приказ у прозору „**Relationships**“.
- **Hide Table** Сакрива изабрану табелу у прозору „**Relationships**“.
- **Direct Relationships** Приказује све релације и повезане табеле за изабрану табелу у прозору **„Relationships“,** уколико оне већ нису приказане**.**
- **All Relationships** Приказује све релације и повезане табеле у бази података у прозору „**Relationships**“. Имајте на уму да скривене табеле (табеле за које је у дијалогу **Properties** потврђен избор у пољу за потврду **Hidden**) и њихове релације неће бити приказане, осим ако је у дијалогу „**Navigation Options**“ потврђен избор у пољу за потврду „**Show Hidden Objects**“.
- **Close** Затвара прозор „**Relationships**“. Ако сте променили распоред у прозору „**Relationships**“, бићете упитани да ли желите да сачувате промене.

ђ) Креирање релације међу табелама

Релацију међу табелама можете да креирате користећи прозор „**Relationships**“ или превлачењем поља из панела **Field List** у лист са подацима. Када креирате релацију међу табелама, није неопходно да заједничка поља имају иста имена, мада је то често случај. Важније је да ова поља садрже исти тип података. Ако је поље примарног кључа поље типа „**AutoNumber**“, поље споредног кључа може бити поље типа „**Number**“ уколико је својство **Field Size** исто за оба поља. На пример, можете повезати поље типа „Аутоматско нумерисање“ и поље типа „**Number**“ ако је својство **Field**

Size за оба поља „**Long Integer**“. Када су оба заједничка поља типа „**Number**“, морају имати исту поставку својства **Field Size**.

Креирање релације међу табелама помоћу прозора „Relationships“

1. На картици **File** кликните на дугме **Open**.
2. У дијалогу **Open** изаберите и отворите базу података.
3. На картици **Database Tools**, у групи **Relationships** кликните на дугме **Relationships**.
4. Ако још увек нисте дефинисали релације, аутоматски ће се појавити дијалог **Show Table**. Уколико се не појави, на картици **Design**, у групи **Relationships** кликните на дугме **Show Table**.

У дијалогу **Show Table** биће приказане све табеле и упити у бази података. Изаберите картицу **Tables** да бисте видели само табеле. Изаберите картицу **Упити** да бисте видели само упите. Изаберите картицу **Оба** да бисте видели и табеле и упите.

5. Изаберите неке табеле или упите, а затим кликните на дугме **Add**. Када завршите са додавањем табела и упита у прозор „**Relationships**“, кликните на дугме **Close**.
6. Превуците поље (обично примарни кључ) из једне табеле у заједничко поље (страни кључ) у другој табели. Да бисте превукли више поља, притисните тастер CTRL, кликните на свако поље, а затим их превуците.

Појавиће се дијалог **Edit Relationships**.

7. Проверите да ли приказана имена поља представљају заједничка поља релације. Ако је име поља нетачно, кликните на име поља и изаберите ново поље са листе.

Потврдите избор у пољу за потврду **Enforce Referential Integrity** да бисте наметнули референцијални интегритет за ову релацију.

8. Кликните на дугме **Create**.

Релациона линија ће бити повучена између две табеле. Ако сте потврдили избор у пољу за потврду **Enforce Referential Integrity**, линија ће изгледати дебље на крајевима. Поред тога, само ако сте потврдили избор у пољу за потврду **Enforce Referential Integrity**, број **1** ће се појавити изнад дебљег дела линије на једном крају релације, а симбол за бесконачно (**∞**) ће се појавити изнад дебљег дела линије на другој страни.

НАПОМЕНЕ

- **Креирање релације један-према-један** Оба заједничка поља (обично су то поља примарног и споредног кључа) морају имати јединствени индекс. То значи да својство **Индексирано** за ова поља треба да буде на постављено на опцију **Да (без дупликата)**. Ако оба поља имају јединствени индекс, Access ће креирати релацију један-према-један.
- **Креирање релације један-према-више** Поље на страни „један“ (обично је то примарни кључ) релације мора имати јединствени индекс. То значи да својство **Индексирано** за ово поље треба да буде постављено на опцију **Yes (no duplicates)**. Поље на страни „више“ *не би* требало да има јединствени индекс. То поље може да има индекс, али мора да дозволи појаву дупликата. То значи да својство **Indexed** за ово поље треба да буде постављено на опцију **No** или **Yes (Duplicates OK)**. Када једно поље има јединствени индекс, а друго нема, Access ће креирати релацију један-према-више.

Креирање релације међу табелама помоћу панела „Field List“

У постојећу табели које је отворена у приказу листа са подацима можете додати поље тако што ћете га превући из панела **Field List**. Окно **Field List** приказује доступна поља у табелама у релацији, као и доступна поља у другим табелама. Када превучете поље из „друге“ (неповезане) табеле, а затим довршите чаробњак за проналажење, нова релација један-према-више ће се аутоматски креирати између табеле у окну **Field List** и табеле у коју сте превукли поље. Ова релација, коју је креирао

Access, не намеће подразумевано референцијални интегритет. Морате да уредите релацију да бисте наметнули референцијални интегритет.

Отварање табеле у приказу листа са подацима

1. На картици **File** кликните на дугме **Open**.
2. У дијалогу **Open** изаберите и отворите базу података.
3. У окну за навигацију кликните десним тастером миша на табелу у коју желите да додате поље и креирате релацију, а затим кликните на дугме **Open**.

Отварање панела „Field List“: Притисните комбинацију тастера ALT+F8. Појавиће се окно **Field List**.

Све друге табеле у бази података окно **Field List** приказује груписане у категорије. Када радите са табелом у приказу листа са подацима, Access приказује поља у две категорије у окну **Field List**: **Fields available in related tables** и **Fields available in other tables**. Прва категорија приказује све табеле које су у релацији са табелом са којом тренутно радите. Друга категорија приказује све табеле са којима ваша табела нема релације.

Када у окну **Field List** кликнете на знак плус (**+**) поред имена табеле, видећете листу имена свих поља која су доступна у тој табели. Да бисте у табелу додали поље, превуците жељено поље из панела **Field List** у табелу у приказу листа са подацима.

Додавање поља и креирање релације из панела „**Field List**“

1. У окну **Field List**, у оквиру **Fields available in other tables** кликните на знак плус (**+**) поред имена табеле да бисте приказали листу поља у тој табели.
2. Жељено поље превуците из панела **Field List** у табелу која је отворена у приказу листа са подацима.
3. Када се појави линија за уметање, отпустите поље на жељену позицију.

Покренуће се **Lookup Wizard**.

4. Следите упутства да бисте довршили **Lookup Wizard**.

Поље ће се појавити у табели у приказу листа са подацима.

Када превучете поље из „друге“ (неповезане) табеле, а затим довршите Lookup Wizard, нова релација један-према-више биће аутоматски креирана између табеле у окну **Field List** и табеле у коју сте превукли поље. Ова релација, коју је креирао Access, не намеће подразумевано референцијални интегритет. Морате да уредите релацију да бисте наметнули референцијални интегритет.

е) Брисање релације међу табелама

Морате да избришете релациону линију у прозору „**Relationships**“ да бисте избрисали релацију међу табелама. Пажљиво поставите курсор тако да показивач буде постављен на релациону линију, а затим кликните на линију. Релациона линија постаје дебља ако је изабрана. Када изаберете релациону линију, притисните тастер DELETE. Имајте на уму да брисањем релације такође уклањате и подршку за референцијални интегритет за ту релацију, уколико је била омогућена. Као резултат тога, Access неће више аутоматски спречавати креирање записа који су сирочићи на страни релације „више“.

1. На картици **Database Tools**, у групи **Relationships** кликните на дугме **Relationships**.

Појавиће се прозор „**Relationships**“. Ако још увек нисте дефинисали релације и уколико први пут отварате прозор „**Relationships**“, појавиће се дијалог **Show Table**. Уколико се дијалог појави, кликните на дугме **Close**.

1. На картици **Design**, у групи **Relationships** изаберите ставку **All Relationships**.

Све табеле које имају релације приказане су са релационим линијама.

2. Кликните на релациону линију која одговара релацији коју желите да избришете. Релациона линија постаје дебља када је изабрана.
3. Притисните тастер DELETE.

– или –

Кликните десним тастером миша и изаберите ставку **Delete**.

4. Access ће можда приказати поруку Are you sure you want to permanently delete the selected relationship from your database?. Ако се појави ова порука о потврди, кликните на дугме Yes.

НАПОМЕНА Релацију нећете моћи да избришете ако друга особа или процес тренутно користи неку од табела употребљених у релацији међу табелама или уколико се нека табела користи у отвореном објекту базе података (као што је образац). Прво морате затворити све отворене објекте који користе ове табеле да бисте могли да избришете релацију.

ж) Промена релације међу табелама

Релацију међу табелама можете променити ако је изаберете у прозору „Relationships“, а затим је уредите. Пажљиво поставите курсор тако да показивач буде постављен на релациону линију, а затим кликните на њу да бисте је изабрали. Релациона линија постаје дебља када је изабрана. Кликните двапут на изабрану релациону линију или на картици **Design**, у групи **Tools** кликните на дугме **Edit Relationships**. Појавиће се дијалог **Edit Relationships**.

Уношење промена у дијалог **Edit Relationships**

1. На картици **Database Tools**, у групи **Relationships** кликните на дугме **Relationships**.

Појавиће се прозор „**Relationships**“. Ако још увек нисте дефинисали релације и уколико први пут отварате прозор „**Relationships**“, појавиће се дијалог **Show Table**. Уколико се дијалог појави, кликните на дугме **Close**.

- На картици **Design**, у групи **Relationships** изаберите ставку **All Relationships**.
2. Све табеле које имају релације приказане су са релационим линијама.
3. Кликните на релациону линију која одговара релацији коју желите да промените. Релациона линија постаје дебља када је изабрана.
4. Кликните двапут на релациону линију.
5. Унесите промене, а затим кликните на дугме **У реду**.

Дијалог **Edit Relationships** дозвољава промену релације међу табелама. Можете да промените табеле или упите са обе стране релације, као и поља са обе стране. Осим тога, можете да подесите тип споја или да наметнете референцијални интегритет и одаберете каскадну опцију.

з) Наметање референцијалног интегритета

Сврха коришћења референцијалног интегритета јесте спречавање појаве записа који су сирочићи и одржавање синхронизованости референци како не бисте имали записе који референцирају друге записе који више не постоје. Референцијални интегритет намећете тако што га омогућавате за релацију међу табелама. Када је референцијални интегритет једном наметнут, Access одбија сваку операцију која би нарушила референцијални интегритет те релације међу табелама. Access одбија ажурирања која мењању одредиште референце и брисања која уклањају одредиште референце.

Укључивање или искључивање референцијалног интегритета

1. У прозору „**Relationships**“ изаберите релациону линију коју желите да промените. Релациона линија постаје дебља када се изабере.
2. Кликните двапут на релациону линију.

Појавиће се дијалог **Edit Relationships**.

3. Потврдите избор у пољу за потврду **Enforce Referential Integrity**.
4. Унесите додатне промене у релацију, а затим кликните на дугме **ОК**.

Када наметнете референцијални интегритет, примењиваће се следећа правила:

- У поље споредног кључа повезане табеле није могуће унети вредност ако та вредност не постоји у пољу примарног кључа примарне табеле – тиме бисте креирали записе који су сирочићи.
- Запис у примарној табели не можете да избришете ако у повезаној табели постоје подударни записи. На пример, не можете да избришете запис о запосленом из табеле

„**Zaposleni**“ ако у табели „**Narudzbine**“ постоје поруџбине које су додељене том запосленом. Међутим, можете да одаберете да избришете примарни запис *и* све повезане записе једном операцијом тако што ћете потврдити избор у пољу за потврду **Cascade Delete Related Fields**.

- Вредност примарног кључа у примарној табели не можете да промените ако ће то довести до креирања записа који ће бити сирочићи. На пример, не можете да промените број поруџбине у табели „**Narudzbine**“ ако постоје ставке додељене тој поруџбини у табели „**Detalji narudzbine**“. Можете, међутим, одабрати да једном операцијом ажурирате примарни запис *и* све повезане записе потврђивањем избора у пољу за потврду **Cascade Update Related Fields**.

НАПОМЕНЕ Ако наилазите на тешкоће у омогућавању референцијалног интегритета, имајте у виду да су следећи услови потребни да би се могао наметнути референцијални интегритет:

- Заједничко поље у примарној табели мора да буде примарни кључ или да има јединствени индекс.
- Заједничка поља морају да имају исти тип података. Једини изузетак је да поље типа „Аутоматско нумерисање“ може да буде повезано са пољем типа „**Number**“ коме је својство **Field Size** постављено на опцију **Long Integer**.
- Обе табеле морају постојати у истој Access бази података. Референцијални интегритет не може да се наметне на повезане табеле. Међутим, ако су изворне табеле у формату програма Access, можете да отворите базу података у којој су ускладиштене и да омогућите референцијални интегритет у тој бази података.

Постављање каскадних опција

Можете наићи на ситуацију у којој имате оправдану потребу да промените вредност на страни „један“ релације. У таквом случају је потребно да Access аутоматски, као део једне операције, ажурира све редове на које промена утиче. На тај начин, ажурирање се у потпуности довршава како база података не би остала у неусаглашеном стању – где су неки редови ажурирани, а неки не. Применом опције „**Cascade Update Related Fields**“, Access вам помаже да избегнете овај проблем. Када наметнете референцијални интегритет и одаберете опцију „**Cascade Update Related Fields**“, а затим ажурирате примарни кључ, Access ће аутоматски ажурирати сва поља која референцирају тај примарни кључ.

Можда ће такође бити потребно да избришете ред и све повезане записе – на пример, запис шпедитера и све повезане поруџбине за тог шпедитера. Из тог разлога Access подржава опцију „**Cascade Delete Related Fields**“. Ако је наметнут референцијални интегритет и одабрана опција „**Cascade Delete Related Fields**“, Access ће аутоматски избрисати све записе који референцирају тај примарни кључ када избришете запис који садржи примарни кључ.

Укључивање или искључивање каскадног ажурирања и/или каскадног брисања

1. У прозору „**Relationships**“ изаберите релациону линију коју желите да промените. Релациона линија постаје дебља када се изабере.
2. Кликните двапут на релациону линију.

Појавиће се дијалог **Edit Relationships**.

3. Потврдите избор у пољу за потврду **Enforce Referential Integrity**.
4. Потврдите избор у пољу за потврду Cascade Update Related Fields или Cascade Delete Related Fields односно у оба.
5. Унесите додатне промене у релацију, а затим кликните на дугме **OK**.

НАПОМЕНА Ако је примарни кључ поље типа „**AutoNumber**“, потврда избора у пољу за потврду **Cascade Update Related Fields** неће имати ефекта зато што не можете променити вредност у пољу типа „**AutoNumber**“.

Пример: Библиотека

ISBN	Naslov	Autori	Nobel	Datum izd.	Izdavac	Zemlja	Broj st.	Cena	Zanr
9780000000001	The Idiot	Fjodor Dostojevski	Ne	14-08-2001	Penguin	SAD	240	14	Klasika
9780000000002	Zločin i kazna	Fjodor Dostojevski	Ne	14-11-2003	Random	Srbija	280	14	Klasika
9780000000003	Growth of the Soil	Knut Hamsun	Da	14-11-2001	Penguin	SAD	260	16	Klasika
9780000000004	Glad	Knut Hamsun	Da	14-02-2002	Prosveta	Srbija	280	18	Klasika
9780000000005	The Kite Runner	Khaled Hosseini	Ne	14-08-2002	Penguin	SAD	320	22	Fikcija
9780000000006	A planine su odjekivale	Khaled Hosseini	Ne	27-04-2004	Laguna	Srbija	340	24	Fikcija
9780000000007	Na Drini cuprija	Ivo Andric	Da	20-08-1945	Prosveta	Srbija	366	50	Istorijski
9780000000008	High Fidelity	Nick Hornby, Mike Hornby	Ne	14-02-2003	Vulkan	Srbija	360	26	Fikcija

Табела 1. Сви подаци које ћемо водити

Следеће табеле су добијено после примене основних принципа дизајна базе података и поступка нормализације.

Табела Књиге:

ISBN	Naslov	Datum izd.	Broj st.	Cena	IzdavacID	ZanrID
9780000000001	The Idiot	14-08-2001	240	14	1	1
9780000000002	Zločin i kazna	14-11-2003	280	14	2	1
9780000000003	Growth of the Soil	14-11-2001	260	16	1	1
9780000000004	Glad	14-02-2002	280	18	3	1
9780000000005	The Kite Runner	14-08-2002	320	22	1	2
9780000000006	A planine su odjekivale	27-04-2004	340	24	4	2
9780000000007	Na Drini cuprija	20-08-1945	366	50	3	3
9780000000008	High Fidelity	14-02-2003	360	26	5	2

Табела Аутори:

AutorID	Autor	Nobel	Biografija
1	Fjodor Dostojevski	Ne	Velikan ruske knjizevnosti…
2	Knut Hamsun	Da	Prvi nobelovac iz radnicke porodice…
3	Khaled Hosseini	Ne	Pisac iz Avganistana sa ….
4	Ivo Andric	Da	Prvi srpski nobelovac…
5	Nick Hornby	Ne	Prvi od brace Hornby….
6	Mike Hornby	Ne	Drugi od brace Hornby….

Табела Књиге_Аутори:

ISBN	AutorID
9780000000001	1
9780000000002	1
9780000000003	2
9780000000004	2
9780000000005	3
9780000000006	3
9780000000007	4
9780000000008	5
9780000000008	6

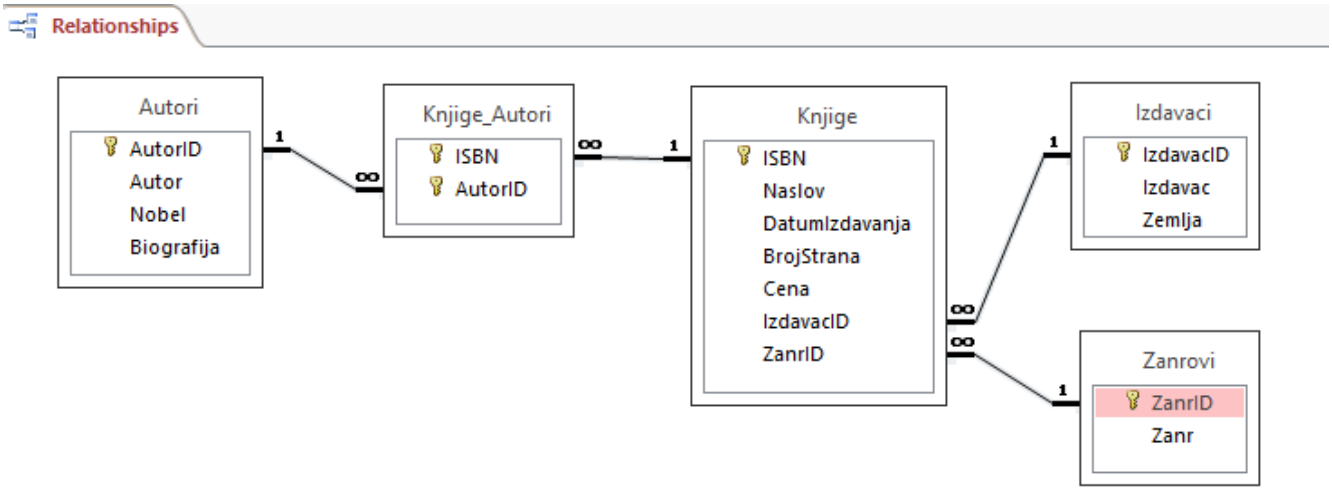
Табела Издавачи:

IzdavacID	Izdavac	Zemlja
1	Penguin	SAD
2	Random	Srbija
3	Prosveta	Srbija
4	Laguna	Srbija
5	Vulkan	Srbija

Табела Жанрови:

ZanrID	Zanr
1	Klasika
2	Fikcija
3	Istorijski

Коначан дизајн базе података Библиотека:



V. Увод у упите (Queries)

Коришћење упита омогућава лакши преглед, додавање, брисање или измену података из саме базе података. Неки од разлога за коришћење упита:

- Брзо проналажење података уз помоћ филтрирања по специфичном критеријуму.
- Израчунавање или сумирање података
- Аутоматизовање процеса управљања подацима, као што је преглед најновијих података

1. SQL (Structured Query Language

SQL jezik је специјално дизајниран језик оптимизован за рад са базама података и манипулацију са подацима.
Неке од најважнијих SQL команди:

- **SELECT** – бира и екстрактује податке из базе података
- **CREATE TABLE** – креира нову табелу
- **UPDATE** – ажурира податеке у бази
- **DELETE** – брише податке из базе
- **INSERT INTO** – убацује нове податке у базу
- **CREATE DATABASE** – креира нову базу

SQL SELECT наредба
Користи се за селекцију података The SELECT statement is used to select data from a database.

- SQL SELECT синтакса

SELECT колана_име, колана_име
FROM табела_име;
или све колоне
SELECT * FROM *табела_име*;
Пример:
SELECT * FROM Knjige;
SELECT Naslov, Cena FROM Knjige;

- SQL SELECT DISTINCT наредба

SELECT DISTINCT наредба се користи да прикаже јединствене (различите) вредности.
SQL SELECT DISTINCT синтакса
SELECT DISTINCT колана_име, колана_име
FROM табела_име;
Пример:
SELECT DISTINCT IzdavacID, Naslov FROM Knjige;

- SQL WHERE клаузула

WHERE користимо да филтрирамо записе по одређеном критеријуму.
SQL WHERE синтакса
SELECT колана_име, колана_име
FROM табела_име
WHERE колана_име оператор вредност;
Пример:
SELECT * FROM Knjige
WHERE ZanrID=1; (све књиге које припадају жанру Класика).

2. MS Access оператори

а) Релацијски оператори

Користе се за поређење вредности и враћа True, False, или Null вредност.

ОПЕРАТОР	СВРХА	ПРИМЕР
<	Мање.	Vrednost1 < Vrednost2
<=	Мање или једнако.	Vrednost1 <= Vrednost2
>	Веће.	Vrednost1 > Vrednost2
>=	Веће или једнако.	Vrednost1 >= Vrednost2
=	Једнако.	Vrednost1 = Vrednost2
<>	Различито.	Vrednost1 <> Vrednost2

б) Логички (Boolean) оператори

Логички оператори комбинују две Boolean вредностии враћају true, false, или null резултат.

ОПЕРАТОР	СВРХА	ПРИМЕР
And	Враћа True кад су оба израза true.	Izraz1 And Izraz2
Or	Враћа True кад је бар један од израза true.	Izraz1 Or Izraz2
Not	Враћа True кад израз није true.	Not Izraz

в) Аритметички оператори

ОПЕРАТОР	СВРХА	ПРИМЕР
+	Сабирање два броја.	[Cena]+[PDV]
-	Одузимање (разлика два броја) или означава негативну вредност броја.	[Cena]-[Popust]
*	Множење два броја.	[Kolicina]*[Cena]
/	Дељење првог броја другим.	[Ukupno]/[BrojProiyvoda]
\	Целобројно дељење.	[Registrovani]\[Soba]
Mod	Остатак целобројног дељења.	[Registrovani] Mod [Soba]
^	Експонент броја.	Broj ^ Eksponent

г) Специјални оператори

ОПЕРАТОР	СВРХА	ПРИМЕР
Like "патерн"	Проверава текстуалне вредности коришћењем џокер знакова: **?**,***** итд.	Polje1 **Like** "program*"
Between Vrednost1 **And** Vrednost2	Проверава да ли је одређена вредност у опсегу.	Polje1 **Between** 1 **And** 10 или Polje1 **Between** #07-01-07# **And** #12-31-07#
In (Vrednost1, Vrednost2...)	Проверава да ли је вредност унутар одређеног сета вредности.	Polje1 **In** ("crvena","zelena","plava") или Polje1 **In** (1,5,7,9)
&	Комбинује два текстуална израза у један.	Tekst1 **&** Tekst2

д) Џокер знаци:

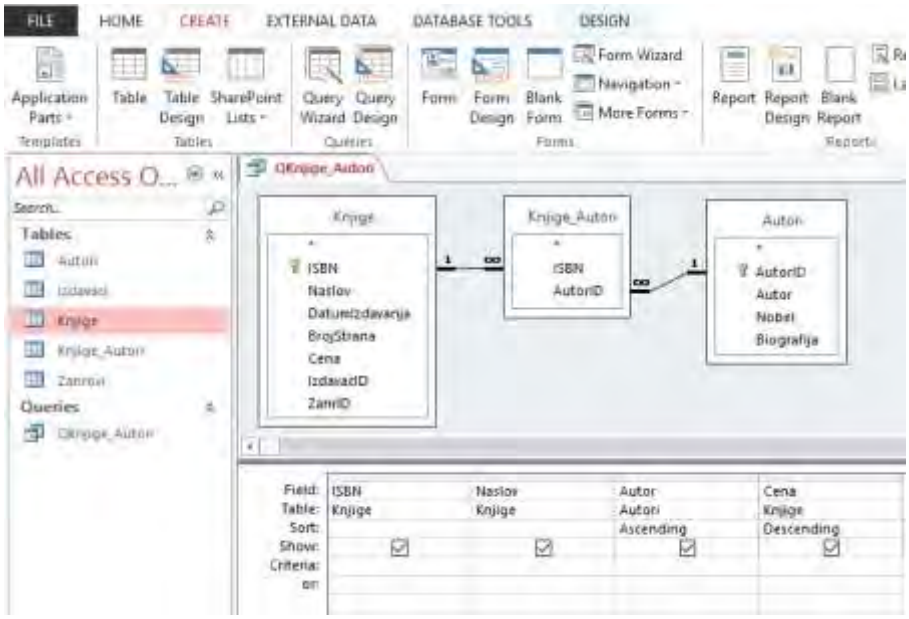
КАРАКТЕР	ОПИС	ПРИМЕР
*	Мења било који број карактера.	**wh*** налази what, white, и why, али не и awhile или watch.
?	Мења један карактер.	**B?ll** налази ball, bell, и bill.
[листа]	Одговара листи карактера унутар заграде.	**B[ae]ll** налази ball и bell, али не и bill.
[!листа]	Одговара листи карактера који нису унутар заграде.	**b[!ae]ll** налази bill и bull, али не и ball или bell.
[-]	Одговара опсегу карактера у растућем редоследу (A до Z, не Z до A).	**b[a-c]d налази** bad, bbd, и bcd.
#	Мења један нумерички карактер.	**1#3** налази 103, 113, и 123.

Демо база података за упите
Користићемо нашу Библиотеку базу података.
Селекција из табеле Књиге:

ISBN	Naslov	DatumIzdavanja	BrojStrana	Cena	IzdavacID	ZanrID
9780000000001	The Idiot	14-Aug-01	240	$40.00	Penguin	Klasika
9780000000002	Zlocin I kazna	14-Nov-03	280	$14.00	Random	Klasika
9780000000003	Growth of the Soil	12-Jul-98	255	$34.00	Penguin	Klasika
9780000000004	Glad	15-Feb-00	188	$24.00	Prosveta	Klasika
9780000000005	The Kite Runner	10-Sep-99	246	$34.00	Penguin	Fikcija
9780000000006	A planine su odjekivale	15-Jun-05	222	$32.00	Laguna	Fikcija
9780000000007	Na Drini cuprija	20-Oct-45	366	$50.00	Prosveta	Istorijski
9780000000008	High Fidelity	23-Nov-06	354	$25.00	Vulkan	Fikcija

3. Креирање упита

На картици Create селектовати Query Design команду. Преко Show Table дијалога изабрати табеле (и/или упите) који ће бити основа новог упита. На примеру доле видимо да су селектоване три табеле: Knjige, Knjige_Autori и Autori. У следећем кораку бирамо поља за упит.
У примеру видимо да су изабрана следећа поља: ISBN, Naslov и Cena из табеле књиге, као и Autor из табеле Аутори. Над пољима Autor и Cena постављене су и опције за сортирање. На следећој слици испод види се резултат извршавања овог упита.



QKnjige_Autori

ISBN	Naslov	Autor	Cena
9780000000001	The Idiot	Fjodor Dostojevski	$40,00
9780000000002	Zlocin I kazna	Fjodor Dostojevski	$14,00
9780000000007	Na Drini cuprija	Ivo Andric	$50,00
9780000000005	The Kite Runner	Khaled Hosseini	$34,00
9780000000006	A planine su odjekivale	Khaled Hosseini	$32,00
9780000000003	Growth of the Soil	Knut Hamsun	$34,00
9780000000004	Glad	Knut Hamsun	$24,00
9780000000008	High Fidelity	Mike Hornby	$25,00
9780000000008	High Fidelity	Nick Hornby	$25,00

А) Упити над једном или више табела

Генерално принцип рада са упитима који укључују једну или више табела је исти: селекција табеле/табела, селекција поља и постављање опција за сортирање и/или филтрирање преко одређених критеријума.

Пример 1: Упит над једном табелом (Књиге): Приказ поља Наслов, БројСтрана и Цена, где наслов почиње једним од следећих слова: БДМН док је цена у распону од 15 до 30.

Field:	Naslov	BrojStrana	Cena
Table:	Knjige	Knjige	Knjige
Sort:			Ascending
Show:	☑	☑	☑
Criteria:	Like "[BDMN]*"		Between 15 And 30
or:			

Пример 2: Упит над више табела: Као и претходни пример уз додатак поља Жанр из табеле Жанрови. Овде се проказују само они записи код којих је број страна већи од 100, жанр почиње великим словом Б, док је цена у распону 20-25 или 100-110.

Field:	Naslov	BrojStrana	Cena	Zanr
Table:	Knjige	Knjige	Knjige	Zanrovi
Sort:				Ascending
Show:	☑	☑	☑	☑
Criteria:		>100	Between 20 And 25	Like "B*"
or:			Between 100 And 110	

Примери критеријума за текстуална поља

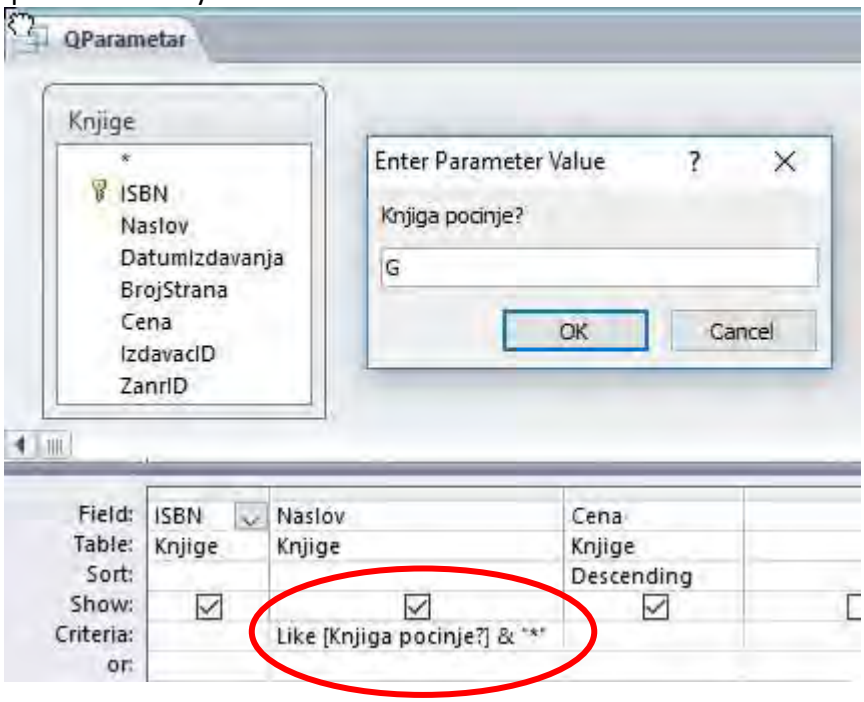
Укључити записе који...	**Критеријум**
Тачна вредност "Glad"	"Glad"
Све осим ове вредности "Glad"	**Not** " Glad"
Почиње на велико слово “G“	**Like** “G*”
Не почиње на велико слово “G“	Not Like “G*”
Садржи део текста “of”	**Like** "*of*"
Завршава се на “ale”	**Like** "*ale"
На другој позицији има слово “l”	**Like** “?l*”
Има 4 карактера	**Like** “????”
Почиње на “G”, “N” или “O”	**Like** “[G;N;O]*”
Не почиње на “G”, “N” или “O”	**Like** “[!G;N;O]*”
Почиње на велико слово у распону од “A” до “M”	**Like** “[A-M]*”
Одговара једној од две вредности "Glad" или "Sreca"	"Glad" Or "Sreca"
Садржи једну од вредности из набројане листе	**In**("Glad"; "Sreca"; "Tuga"; "Sunce")

Примери критеријума за нумеричка поља

Укључити записе који...	**Критеријум**
Тачна вредност 25	25
Све осим ове вредности 25	**Not** 25
Мање или мање једнако 25	< 25 или <= 25
Веће или веће једнако 25	>25 или >=25
Садржи једну од две вредности 20 или 25	20 or 25
Садржи вредност у опсегу	>25 **And** <35 -или- **Between** 25 **And** 35
Вредност ван опсега	<25 or >35
Једну од вредности из набројане листе	**In**(20, 25, 30)

Б) Упити са параметрима

Када се поставе средње заграде у изразу у делу критеријума, онда Access приказује при извршавању упита дијалог за прихват вредности параметра. Следећи пример користи параметар за унос почетног дела наслова књиге, и у комбинацији са Like и & оператором за спајање текста формира флексибилни упит.



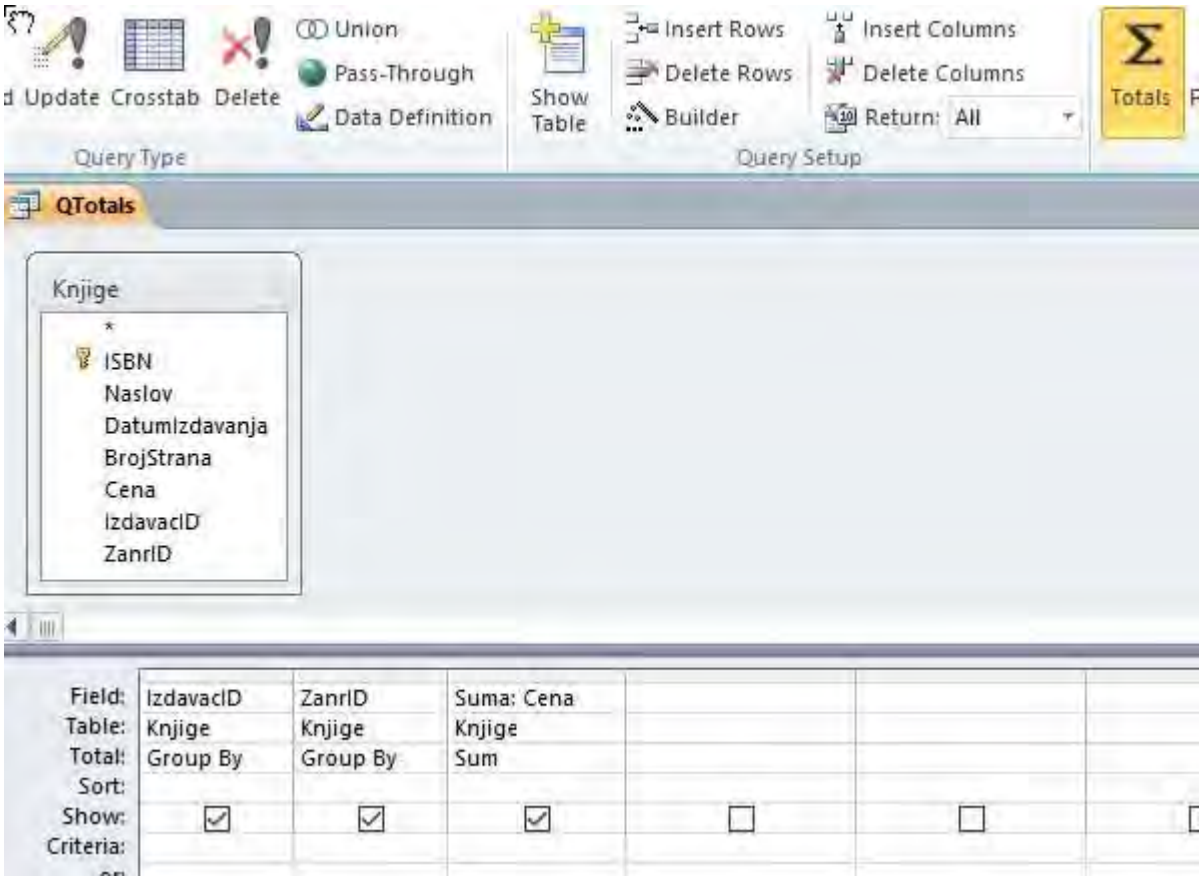
Резултат извршавања претходно приказаног упита, где је корисник унео велико слово Г приказује све књиге чији наслов почиње тим словом.

QParametar

ISBN	Naslov	Cena
9780000000003	Growth of the Soil	$34.00
9780000000004	Glad	$24.00

B) Упити за групна сумирања преко опције Totals

Укључивањем опције Totals на линији алата, појавиће се испод Table реда Total ред у дизајнеру упита. Сва поља за која ћемо радити сумирање треба поставити на Group By, а поље које има вредност која нас интересује се поставља на неку од уграђених функција као што су Sum, Avg, Min, Max или Count које рачунају суму, просек, минимум, максимум или пребројавање респективно. Следећи пример приказује укупне цене груписане по издавачима и жанровима.



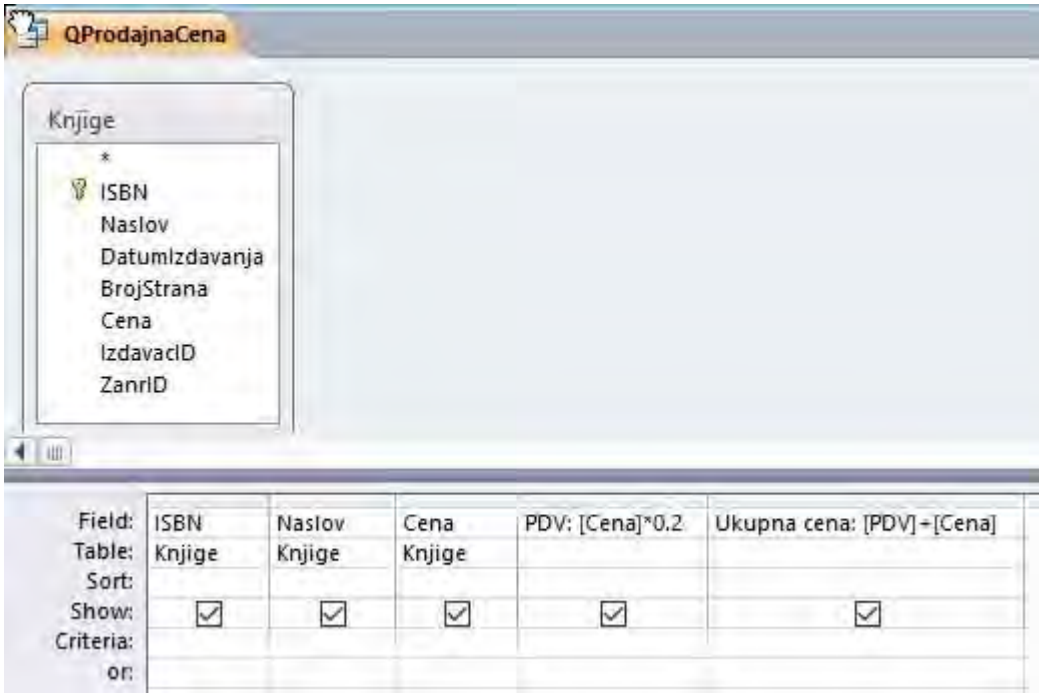
Резултат извршавања претходног упита.

QTotals

IID	ZID	Suma
Penguin	Klasika	$74.00
Penguin	Fikcija	$34.00
Random	Klasika	$14.00
Prosveta	Klasika	$24.00
Prosveta	Istorijski	$50.00
Laguna	Fikcija	$32.00
Vulkan	Fikcija	$25.00

Г) Упити са израчунатим пољима:

Када на реду Field почнемо са следећом синтаксом **ImeIzracunatogPolja:** ушли смо конструисање израза за рачунање вредности израчунатог поља. Следећи пример рачуна вредности пореза на додату вредност (20% од цене), као и укупну цену (збир цене и вредности ПДВ-а.

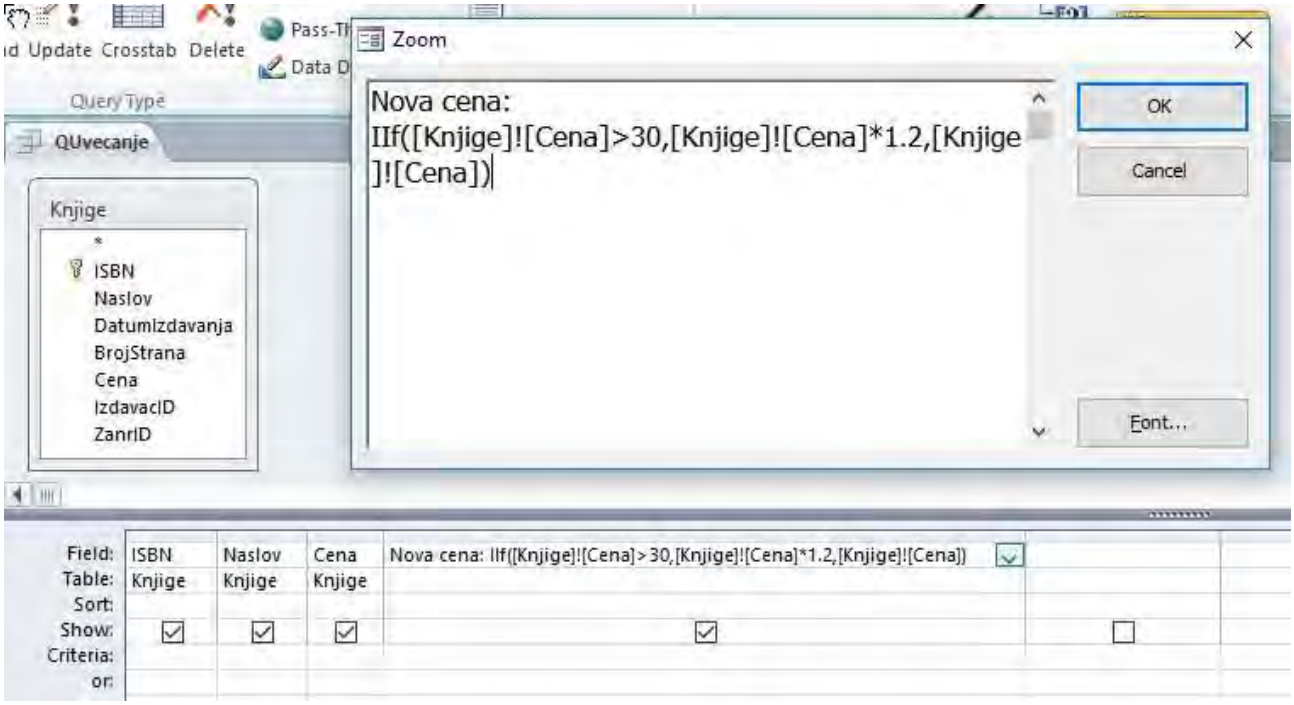


Резултат извршавања претходног упита.

QProdajnaCena

ISBN	Naslov	Cena	PDV	Ukupna cena
9780000000001	The Idiot	$40.00	$8.00	$48.00
9780000000002	Zlocin I kazna	$14.00	$2.80	$16.80
9780000000003	Growth of the Soil	$34.00	$6.80	$40.80
9780000000004	Glad	$24.00	$4.80	$28.80
9780000000005	The Kite Runner	$34.00	$6.80	$40.80
9780000000006	A planine su odjekivale	$32.00	$6.40	$38.40
9780000000007	Na Drini cuprija	$50.00	$10.00	$60.00
9780000000008	High Fidelity	$25.00	$5.00	$30.00

Следећи пример користи уграђену функцију Iif, где све књиге које су скупље од 30 апоена додатно поскупљују за 20%. Општи облик Iif функције је следећи: Iif (Услов, ВредностЗаИстину, ВредностЗаНеистину). Пример такође користи Zoom опцију ради лакшег писања израза.



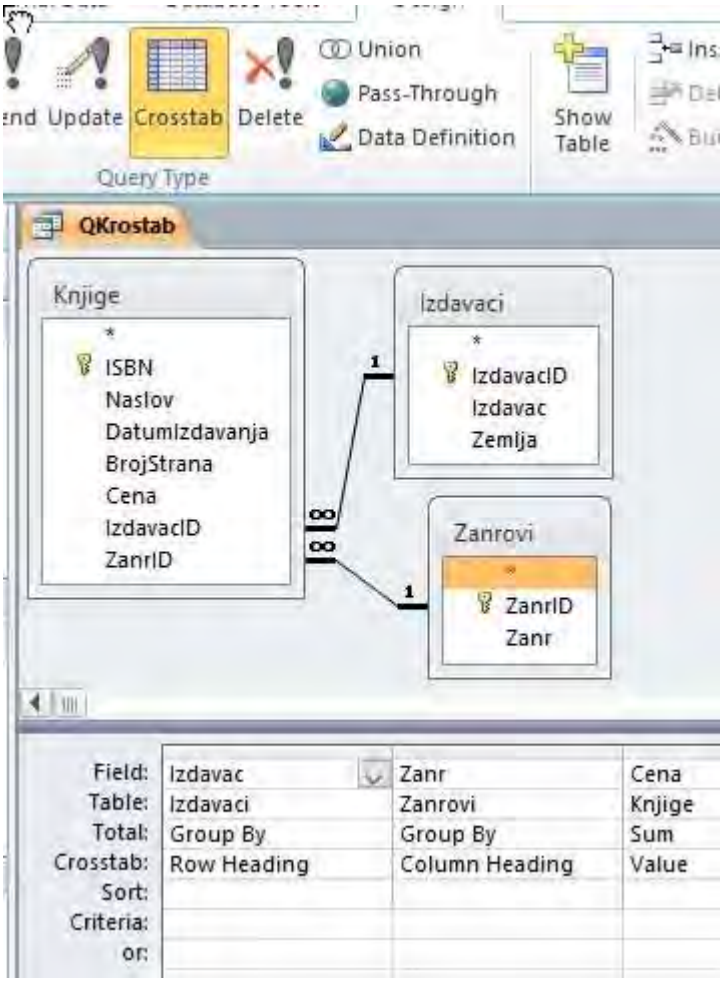
Резултат извршавања претходног упита.

ISBN	Naslov	Cena	Nova cena
9780000000001	The Idiot	$40.00	$48.00
9780000000002	Zlocin i kazna	$14.00	$14.00
9780000000003	Growth of the Soil	$34.00	$40.80
9780000000004	Glad	$24.00	$24.00
9780000000005	The Kite Runner	$34.00	$40.80
9780000000006	A planine su odjekivale	$32.00	$38.40
9780000000007	Na Drini cuprija	$50.00	$60.00
9780000000008	High Fidelity	$25.00	$25.00

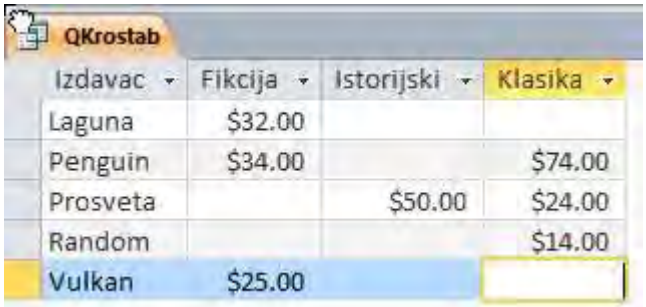
Д) Унакрсни упити (Crosstab Queries)

Сви досада приказани примери упита су били типа Select. За овај пример морамо да уместо Select изаберемо Crosstab тип на линији алатки, као што је приказано на слици. Crosstab упит мора да има минимално три поља, од којих је једно Row Heading, друго Column Heading, а треће поље је вредносно. Следећи пример приказује укупну вредност (суму свих цена) књига по издавачима и жанровима.

Практично идентично претходном примеру са сумирањем preko opcije Totals, само што је приказ резултата у другом формату.



Резултат извршавања унакрсног упита.



Izdavac	Fikcija	Istorijski	Klasika
Laguna	$32.00		
Penguin	$34.00		$74.00
Prosveta		$50.00	$24.00
Random			$14.00
Vulkan	$25.00		