Факултет инжењерских наука Универзитета у Крагујевцу

Назив студијског програма: Машинско инжењерство
Ниво студија: Основне академске студије
Модул: Информатика у инжењерству
Предмет: Базе података
Број индекса: 30/2015

Никола З. Николић

Објектно-релационе базе података

завршни рад

Комисија за преглед и одбрану:

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

Датум одбране:_____

Оцена:_____

У оквиру завршног рада са темом "*Објектно-релационе базе података***" кандидат треба да:**

Проучи литературу и презентира основе објектно-релационих база података. Целине завршног рада треба да обухвате уводна разматрања, кратак приказ релационих и објектно оријентисаних база података, упоредну анализу релационих, објектних и објектно-релационих модела база података. На крају рада дати закључна разматрања.

Препоручена литература:

1. Лазаревић Б. ….: ***Базе података***, ФОН Београд, Београд 2003.
2. Павловић-Лажетић Г.: ***Основе релационих база података***, Математички факултет, Београд, 2000.
3. Melton J.: ***Advanced SQL: 1999 - Understanding Object-Relational and Other Advanced Features***, Morgan Kaufmann Publichers, 2002.
4. Stonebraker M., Moore D: ***Object-Relational DBMSs: The Next Great Wave***, Morgan Kaufmann Publichers, 1996.

Крагујевац, јун 2018.

Ментор:
Др. Милан Ерић

Prof. Dr Milan D. Erić

Садржај

Резиме

Рад даје информације о тренутно најкоришћенијим моделима база података, а то су :релациони, објектни и објектно-релациони модел..Разматрају се карактеристике,предности и мане датих модела.Посебна пажња се поклања објектно-релационом моделу, предностима и манама, као и поређења модела међусобно.Информације у раду пружају кориснику знање за добар одабир модела за одређену апликацију.Изложени су примери који нам помажу да боље схватимо концепт одређеног модела.

***Кључне речи**:* релациони модел, објектни модел, објектно-релациони модел, релација, објекат.

Abstract

The paper gives information on the currently most used models of databases, relational, object and relational-object model. The characteristics, advantages and disadvantages of given models are analyzed. We pay special attention to the object-relational model of advantages and disadvantages, as well as modeling of the models with one another.
The information provided in the workplace gives the user the knowledge of a good model selection for a specific application. There are examples that help us to better understand the concept of a database model.

Keywords: relational model, object model, object-relational model, relation, object.

Увод

Рад се бави проширеним релационим базама података или објектно-релационим моделом.Посебна пажња се поклања разликама између објектног,релационог и објектно-релационог модела.Са развојем објектно-оријентисаног програмирања апликације траже нове моделе база података који би подржали нови концепт у програмирању.Идеја заговорника релационог модела јесте била да се постојећи модел прошири неким новим концептима који би могли да подрже нове захтеве на тржишту.Ова проширења стварају нови модел објектно-релациони модел који задржава релационе табеле али нарушена су нека правила која се тичу самог концепта релација.У поглављима ће бити дате битне разлике ова два система, а биће говора и о објектном моделу који је имао утицај на објектно-релациони модел.На крају ће се донети закључак који се тиче функционалности ова три модела (релациони , објектни и објектно-релациони) на одређеном примеру.Биће представљене мане и предности сваког модела појединачно.Овиме се пружа да пројектанти информационих система изаберу модел са најбољим перформансама за њихову апликацију.

1.Модел података

1.1 Појам и врсте модела података

Као што смо већ споменули рад даје информације о објектно-релационим базама података.Из самог наслова се закључује да се састоји из комбинације нека два модела тј. добрих страна два споменута модела, а то су објектни модел и релациони модел података. Пре самог описивања ова два модела битно је направити неки увод о самом појму модела података.Најпознатији модели су :

1. Хијерархијски модел
2. Мрежни модел
3. Релациони модел
4. Објектни модел

За нас су важни релациони и објектни модели података које ћемо разматрати детаљније у наредним поглављима. Примарни циљ модела података у контексту база податак јесте да обезбеди формални систем за представљање података и манипулисање подацима у бази података.[1].Када разматрамо појам модела можемо га поделити на три важна сегмента која га чине а то су :

а) Скуп типова објеката

б) Скуп операција над објектима дефинисаних типова

в) Скуп правила интегритета

а) Први сегмент јесте структурни део модела.То заправо значи да скуп типова објеката (типовима објеката описујемо ентитете) чини структурни део модела.На пример, имамо типове ентитета РАДНИК, ОДЕЉЕЊЕ.Сваки од ентитета има своја њима важна својства.Радник тако има шифру радника, име, презиме ...Одељење пак има своја обележја број одељења, назив...

Слика 1 Типови ентитета базе података неке фирме

У зависности од модела које смо горе навели (хијерархијски, релациони, мрежни) изабраћемо начин описивања ових типова ентитета са теоријом која стоји иза изабраног модела на тај начин да нам што боље опише дати ентитет. [1]

б) Скуп операција над објектима дефинисаног типа претставља део модела који се бави манипулацијом ентитетим.Када кажемо манипулација ентитетима мислимо на операције које вршимо над њима.Ту спадају мењање података (ажурирање базе), претраживање, уклањање из базе итд. Вратимо се на пример који смо поменули, раднике и одељење.Можемо додати новог радника, пронаћи радника са највећим бројем радних сати итд. [1]

в) Скуп правила интегритета се односе на интегритетни део модела. Правила интегритета дефинишу услове интегритета података у бази података, тј. својства података или односа међу подацима, који морају бити задовољени да би стање базе било ваљано. [1]
Да би смо разумели претходну дефиницију која се односи на правила која владају у моделу показаћемо неке од могућих примера интегритета појединачних ентитета.

- Један радник је пријављен на једном одељењу
- Свако одељење има више од 10 радника али не више од 20
- Називи одељења морају бити различити

Ово су били основни појмови и разматрања када је у питању појам модела података сада ћемо да се вратимо на врсте тих модела, односно описаћемо укратко хијерархијски модел и мрежни, а потом и нама најважнији релациони и објектни.

1.2 Хијерархијски модел

Сам назив на неки начин одаје организованост овог модела ,а он је такав да је у облику стабла.Подаци су смештени у ентитете, при чему се ентитет на највишем хијерархијском нивоу назива, тзв. корен.Ентитети са виших и нижих нивоа су међусобно повезани релацијама. [1]

1.3 Мрежни модел

Због саме природе и ограничености хијерархијског модела јавила се потреба за настанком новог модела података такозваног мрежног модела.Ако би се сагледале карактеристике овог модела видело би се да је он заправо варијација хијерархијског модела.Овакав модел се заснива на мрежи података који немају надређени односно подређени ентитет.У оваквом моделу могуће је ући са било ког чвора. [1]

2. Релациони модел

Пошто смо направили увод и разјаснили битне сегменте модела покушаћемо да сада релациони модел провучемо и детаљније објаснимо кроз сва три сегмента, а то су : скуп типова објеката, скуп операција над објектима дефинисаних типова и скуп правила интегритета.Сада ће бити лакше да се ово појединачно објасни јер смо већ разјаснили сваки појам понаособ у претходном поглављу. [1]

2.1 Структурни део релационог модела

Релациони модел за управљање базама података је једна од врста модела коју структурно карактеришу два основна типа објеката, а то су релација и домен.Домен функције је у математици скуп на коме је дата функција дефинисана.У релационом моделу домен представља скуп вредности који су истог типа. Ако би се то представило примером радника и одељења, вредности истога типа би биле имена радника , презимена радника, скуп назива одељења.Ради бољег разумевања изречених дефиниција погледајмо слику 2.

Слика 2 Изглед релације РАДНИК

Домен је једноставан ако су му све вредности атомичне, то значи да систем за управљање може једнозначно да пронађе домен.Уколико је домен састављен из више компонената тада се ради о сложеном или композитном домену.Пример би био Адреса.За овај податак морамо знати Улицу, Место, број, дакле ради се о сложеном домену.Најчешће коришћени домени су скуп целих бројева (у програмирању „integer“), скуп децималних бројева (double), низ алфанумеричких знакова (string), датум.У нашем примеру а на слици 2 јасно можемо видети да су сви атрибути релације једноставни па можемо имати овакав табеларни приказ, ово још значи да се релација налази у првој нормалној форми или скраћено 1NF.Дакле да резимирамо колоне табеле одговарају атрибутима релације, а врсте представљају н-торке релације. [1]

Због овакве организованости табела има следеће особине а то су :

1. Нема дуплирања врста
2. Редослед врста је небитан
3. Редослед колона је небитан
4. Све вредности у табели су атомичне

Сада ћемо показати како се записује релација R.Ознака R Представља релацију, у даљем тексту A ће представљати атрибут док је са великим словом D обележен домен.Ево како би тај запис изгледао : *R*(*A*1 : *D*1*; A*2 : *D*2*; ... ;An* : *Dn*).Атрибут A1, релације R, узима вредност из домена D1, A2 узима вредност из домена D2 и тако надаље докле год имамо атрибута.Да поновимо ознака R представља ознаку релацијске шеме. [1]

Релациона база података је скуп скупова података који се на логичком нивоу могу посматрати као нормализоване релације одговарајућих степена и динамичког садржаја. Скуп релацијских шема релационе базе података је шема релационе базе података. Базне релације су релације које су дефинисане независно од других релација у бази података, и које се не могу извести из других базних релација. Оне су на физичком нивоу представљене одговарајућим структурама података. Изведене релације су релације које се могу извести из базних релација применом скупа операција. Оне обично немају физичку репрезентацију, и представљају различите погледе које корисници могу имати на исту базу података. [1]

Неке друге дефиниције релационог модела не укључују појам домена у основне типове објеката структуре релационог модела. Међутим, овај појам има свој операциони значај у томе што домени одређују скуп операција које се над њиховим вредностима могу извршавати. Тако се вредности атрибута над истим доменима могу поредити (а вредности атрибута над различитим доменима не могу), па се над атрибутима, дефинисаним над истим доменима, могу извршавати и све операције које укључују поређење. Вечина упитних језика, као што је SQL, не познаје семантички аспект домена, па је могуће, на пример, постављати упите којима се пореде наслови књига са именима држава.[1]

Пошто сада ово знамо написаћемо шеме релација за пример радника и одељења.

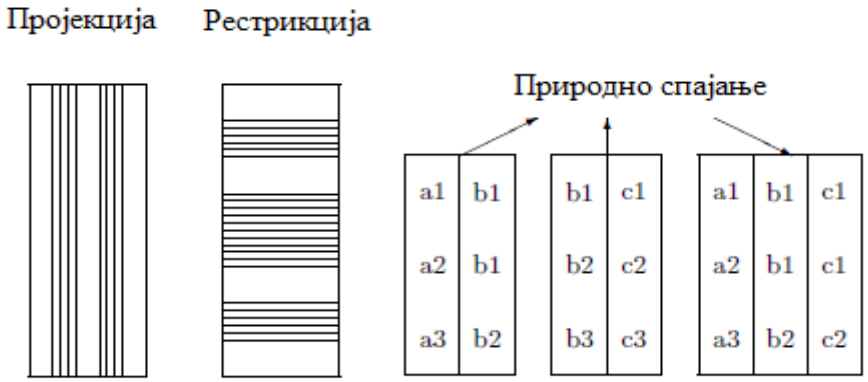
RADNIK (SIF_RADNIKA, IME, PREZIME)
ODELJENJE (SIF_ODELJENJA, NAZIV)

Ове две релације представљају типове ентитета радник и одељење.У заградама као што је већ објашњено се налазе атрибути шема релације.

2.2 Манипулативни део релационог модела

Манипулативни део релационог модела представља формализам за дефинисање релационог израза општег облика, чија је вредност произвољни подскуп скупа података из базе.[1] Постоје два формална језика који раде са релацијама, то је релациона алгебра и релациони рачун.Најједноставније како објашњавамо релациону алгебру јесте да је она скуп операција над релацијама. [1] Изрази који манипулишу релацијама се састоје из низа операција над тим релацијама.Операције које се извршавају том приликом су :

- Пројекција
- Рестрикција
- Природно спајање



Слика 3. Операције : пројекције, рестрикције и природног спајања

2.3 Рестрикција

Сада ћемо мало објаснити ове најважније операције,почећемо од рестрикције.На слици 3 је то други графички приказ како је и наглашено.Овде се графичк види фиктивно налажење н-торки које одговарају задатом изразу то јест као резултат дају одговарајући скуп за задату алгебру што је и сврха операције рестрикције.Представљени скуп је приказан хоризонталним линијама.

На пример, Ако имамо релацију R која има два атрибута X и Y и ako је Θ оператор (може бити <, >, = итд.) постављен тако да X Θ Y дефинисан израз и да може да процени истинит израз, тада је Θ рестрикција релације R. Укратко оператор рестрикције релације на све атрибуте X и Y има исто заглавље као R али садржи само оне торке које задовољавају задати услов.У пракси би ово било : Наћи све раднике који имају име „Иван“. [1]

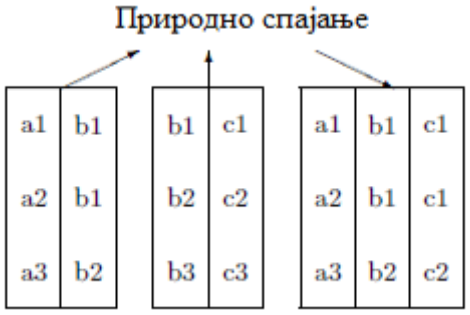
‘Radnik WHERE ime = Ivan' – изглед израза у језику SQL

2.4 Пројекција

За разлику од рестрикције пројекцијом представља налажење атрибута релације који одговарају датом изразу.Усправне линије са слике 3 графички илуструју овај процес.
Нека релација R која атрибуте X , Y ... Z.Тада се пројекција релације R на X , Y ... Z означава са R {X, Y, ..., Z} и представља релацију чије је заглавље изведено из R уклањањем свих атрибута који не одговарају задатој алгебри релације.Торке се том прилик јављају и чине тело нове релације где важи {X:x, Y;y, ..., Z;z}.[1]

2.5 Природно спајање

Природно спајање представља операцију спајања релација по једнакости појединачних атрибута две релације, или по једнакости подскупова одређених атрибута.



Слика 4. Природно спајање

Нека имамо релације R и S са скуповима атрибута $\{A_1, A_2, \ldots, A_n\}$ и $\{B_1, B_2, \ldots, B_m\}$.Нека је X подскуп атрибута релације R, а Y подскуп атрибута релације S. Нека је Z разлика између X и скупа атрибута прве релације R, а W разлика између Y и скупа атрибута друге релације S. Релације R и S се тада могу представити као R[X,Z] и S[Y,W]. Тада се операција спајања релација R и S по скуповима атрибута X,Y дефинише на следећи начин: Релације R и S се тада могу представити као R[X,Z] и S[Y,W]. Тада се операција спајања релација R и S по скуповима атрибута X,Y дефинише на следећи начин:

$$R[Y \div Z]T = \{x | x \in R[X] \wedge \{x\} \times T(Z) \subseteq R(X,Y)\}$$ [1]

Ово су биле неке основне операције релација поред описаних операција постоје дељење, унија, пресек, разлика, декартов производ.Ово је било потребно објаснити како би смо схватили сврху релационе алгебре а она је ту да бисмо :

- Дефинисали простор за дохватање података
- Дефинисање простора за ажурирање података
- Дефинисање правила интегритета
- Дефинисање изведених релација

Постоји још разлога зашто је било потребно увести алгебру у свет база података и информационог система, без које би исти били не могући.

2.6 Релациони модел интегритетни део

Интегритет као појам би значио нешто свеукупно, потпуно, не дељиво.У релационом моделу, тј. интегритет ове врсте модела представља услове које треба да задовоље подаци да би база података била прихватљива.Ови услови морају бити испуњени у сваком тренутку.Систем за управљање базама података при свакој промени проверама задате услове интегритета.Ово можемо да схватимо као тврдње или неке истините формуле које морају бити тачне у свакој ваљаној бази података.Поставља се питање на ком нивоу,то јест како и где ове тврдње и истините формуле делују, на шта се могу односити? Одговор би био да се услови интегритета могу односити на појединачне атрибуте, домен или релацију, као и на целу базу података. [1]
Услови интегритета се могу односити на атрибут, релацију, базу података и тада су специфични при чему се задају експлицитно, или општи услови који се могу применити на сваку базу података или релацију, и који су зато имплицитни.[1]
Специфични услови интегритета могу бити двојаки. Прва врста тих услова односи се на карактеристике појединачних атрибута или њихових домена, или на везу између специфичних вредности атрибута н-торки једне или више релација.На слици 5 је дата табела „Osoba“са атрибутима. [1]

Osoba

JMBG	IME	DATUM_RODJENJA
1709997740017	NIKOLA	17.09.1999
1711988740057	BOJAN	17.11.1988
2704998640047	JOVAN	27.04.1998
0701996840057	MARIJA	07.01.1996

Слика 5. Релација „ Особа“, табеларни приказ релације

Дакле имамо табелу „Osoba“ са атрибутима „JMBG“ , „IME“ и „DATUM_RODJENJA“.Види се са слике да у табели важе нека правила то су управо интегритетна правила која се односе на атрибуте, релацију итд.Тако да можемо уочити да :

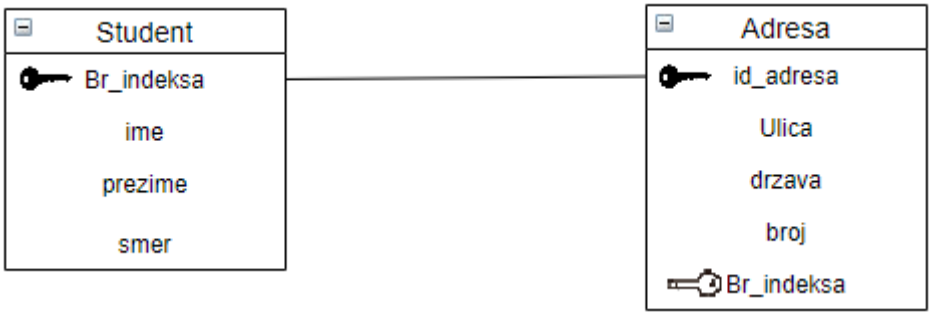
- Свака вредност из домена атрибура „JMBG“ мора имати тачно тринаест бројева који представљају јединствени матични број особе.
- Бројеви морају бити цели (У програмирању „Integer“)
- Име може да садржи до тридесет карактера
- Датум рођења особе мора бити типа („date“) датумског записа дан, месец и година рођења
- Сва поља морају имати вредност (нпр. особи се несме изоставити „JMBG“)

2.7 Примарни кључ

Појам примарног кључа се у релационом моделу може дефинисати као кључ који јединствено индентификује било коју врсту у табели.На основу примарног кључа једне релације могуће је дефинисати концепт страног кључа друге релације.

2.8 Страни кључ

Ако нам је познат кључ једне релације могуће је дефинисати концепт страног кључа друге релације.Представља поље у табели које је повезано са примарном табелом, на тај начин што чува примарни кључ табеле са којом је повезана као страни кључ.

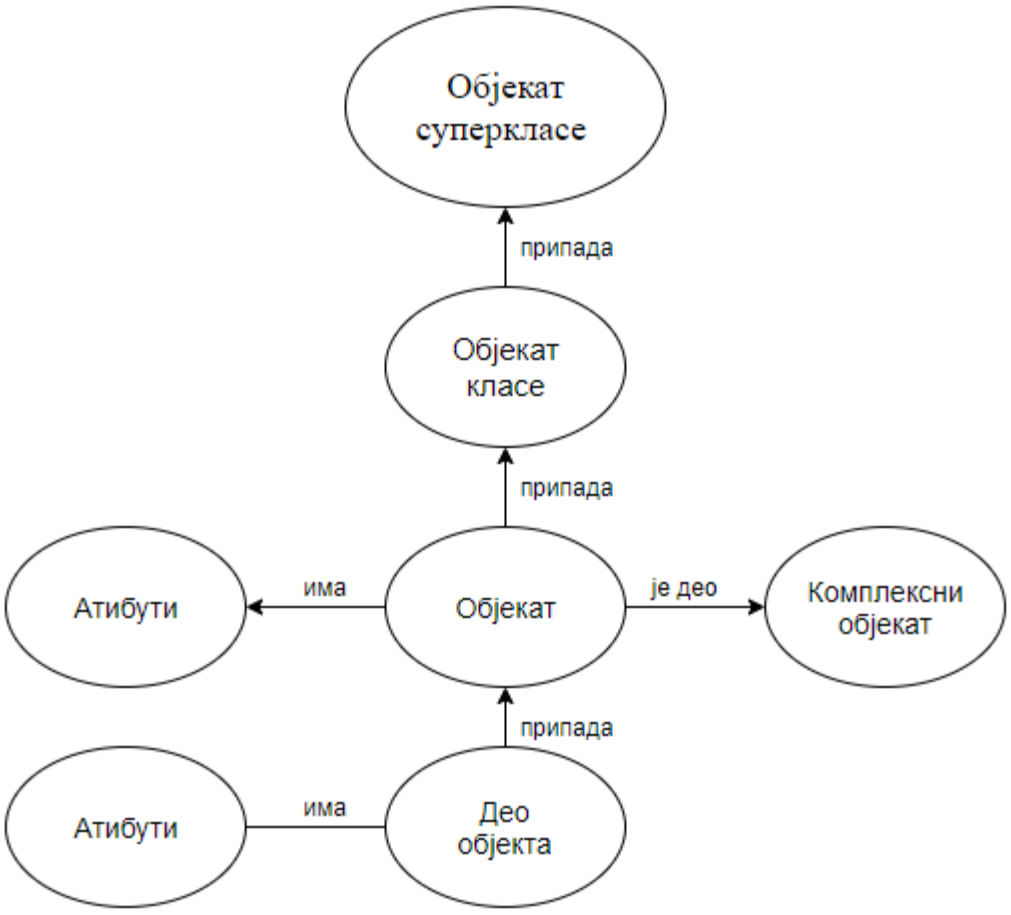


Слика 6 повезивање две релације

У релацији „Student“ индентификатор релације је атрибут „Br_Indeksa“ који се у релацији „Adresa“ налази у улози страног кључа.Тиме смо повезали две релације.

3. Објектно орјентисани модел

До сада се говорило о релационом моделу, тај модел је био од значаја јер је унапредио ову област што се са хијерархијским и мрежним моделом није могло рећи.Овај напредак довео је до развоја непроцедуралних упитних језика и у значајној мери допринео независности података.Развој и предност коју је донео релациони приступ базама је неоспорив, међутим развојем програмских језика класа, објектно орјентационог програмирања довео је до тога да се проналазе неки нови модели који ће моћи да подрже моћне апликације са великом структуром и великим избором типова података.Једна од мана релационог модела је управо та да има оскудан избор типова података, као и захтеви по питању нормализације релација. Због ових мана апликације као што су „Computer Aided Design – CAD),“ која ради пројектовање за нас,географски информациони системи, хипертекстуалне апликације, као и апликације развоја софтвера помоћу рачунара „Computer Aided Software Engineering– CASE“ су захтевале нови вид чувања података прилагодљив потребама апликација као и већим диапазоном типова податак.Ово је довело до развоја објектно-оријентисаног модела база података без којег ове апликације не би биле онакве какве их данас познајемо. [1]



Слика 7 Модел објектно оријентисане базе података

Ова промена је била постепена то јест релационе базе најпре добијају проширење то јест уживају потпуно новом функционалношћу, као што је :

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

Ова проширења доводе до изласка из оквира стандардног релационог система и прелазе у систем нове генерације.

Базе података изграђене над објектно-оријентисаним моделом су објектно-оријентисане базе података, а систем за управљање таквим базама је „DDL“ и „DML“, по аналогији са релационим и дистрибуираним системима, објектно-оријентисани систем за управљање базама података, у ознаци OOSUBP. Објектно оријентисане базе података, заједно са OOSUBP, образују систем објектно оријентисаних база података. За обе врсте система користи се и краћи термин, објектни систем, када је из контекста јасно на коју врсту система се термин односи. [1]

Управо смо увели нови концепт модела у базама података који сада има већу подршку и прилагодљивији је новим методама у програмирању пре свега мислимо на објектно-орјентисано програмирање са великом структуром класа и објеката, међутим пред овим системом OOSUBP (Објектно-орјентисани систем за управљање базама података) се постављају слични захтеви као и пред RSUBP (Релациони систем за управљање базама података) а то су :

- Упитни језик
- Интегритетни део
- Управљање трансакцијама
- Физичка организација
- Индекси и груписање објеката
- Конкурентност и ауторизација

Иако је овакав систем компликованији за пројектовање комплексних програмских система он доживљава велику популарност.Релациони модел има јасну дефиницију, тј. лакше га је схватити и дефинисати док је објектни модел доста незахвалан по том питању.Концепти на којима објектни системи леже су доста различити.

Без обзира на одсуство јединства у концептима и стандарда у моделу, архитектури и језицима, постоји већи број прототипских и комерцијалних ООСУБП, нпр. GemStone , ObjectStore , Iris , ORION , O2 , итд. Како се функционалност проширених релационих система преклапа (а у неким системима и поклапа) са објектним системима, и они први се могу сматрати објектним системима.
(нпр. POSTGRES, Starburst). [1]

Модел података представља логичку репрезентацију објеката реалног света и њихових односа, могућности манипулисања објектима и задавања правила интегритета.Основна идеја објектно оријентисаног модела је да подигне ниво апстракције података, тако да се, уместо битовима, бајтовима, слоговима, пољима, н-торкама, манипулише природнијим ентитетима из реалног света, објектима. [1]

У претходном поглављу се говорило да јединствени објектни модел не постоји.
Заговорници овог модела одустају од намере за дефинисањем објектно орјентисаног модела уз следећи став :
"У односу на спецификацију (ОО) система примењујемо дарвинистички приступ: надамо се да ће се, из низа експерименталних прототипова који су у изградњи, искристалисати јединствени објектно-оријентисани модел" [5].

Пошто су се објаснили недоумице око дефинисања и изостанка неког јединственог модела , морају се препознати неки минимални стандарди тј. минимали скуп објектних концепата које на неки начин сваки објектни систем подржава и по коме је препознатљив, да би смо могли да препознамо и побољшамо неком новом надградњом тај модел.

3.1 Језгро објектног модела

Прво што ће се урадити јесте да се пронађе нешто заједничко за све објектне моделе.
То су неки минимални захтеви које ћемо објаснити како би нам овај модел био што јаснији.

Готово сваки објектно-орјентисани модел података садржи следеће концепте без којих не би ни био објектни, а то су :

- Класа
- Објекат
- Порука и метода
- Учаурење
- Наслеђивање

Утврђивањем минималних захтева објектно орјентисаног модела,може се прећи на појединачно описивање.[1]

3.2 Објекат

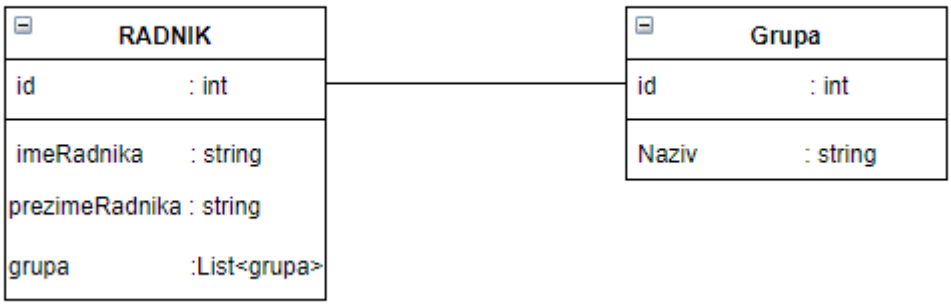
Када се говори о објектно-орјентисаном моделу података објекат представља сваки ентитет.Овај модел под објектом подразумева све.Ово значи да се објектом назива и нека вредност нпр. неки број 48 или нека вредност „ссс“.Објекти су и комплексни ентитети типа „Особа“.Сваки од ових објеката да би се разликовали придружује им се и идентификатор.Рецимо некој особи типа „Особа“придружимо и идентификатор O1.
Идентификатор има и одређених улога у објектним базама.
Сваки објекат има :

- Стање објекта
- Понашање објекта

Стање објекта описује се вредностима његових променљивих. Улога јединственог идентификатора објекта у објектно оријентисаним базама је да замени објекат када се он појави као вредност неке инстанцне променљиве неког другог објекта. [1]
Следи пример објекта и идентификатора.

Ime : Petar
Prezime : Petrovic
Datum_rodjenja : 15.03.1994.
Pohadja : t1

Примећује се да у овом објекту као вредност за „Pohadja“ (Факултет или школа) стоји идентификатор „t1“ који може да садржи своје вредности.Ово је јако олакшавајуће и лакше записати него да смо записали у објекту којег имамо све вредности другог објекта.Објекат под „t1“ може бити школа или факултет који имају вредности нпр. Назив, место, адреса ...



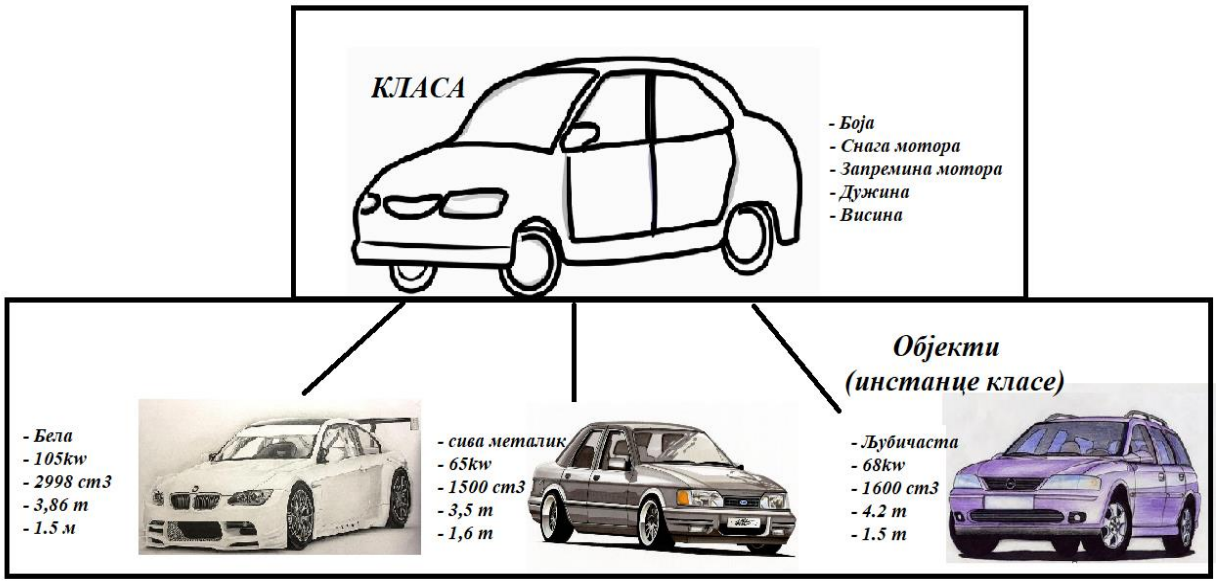
Слика 8 Пример објектног модела

3.3 Порука и метода

Понашање објекта описује се скупом порука на које је објекат способан да одговори. За објекат o1 те поруке могу бити, између осталих, име, године старости, назив послодавца. Објекат на примљену поруку одговара извршавањем методе, тј. имплементиране процедуре која је задужена за "извршавање" поруке. Сваки објекат је учаурен, што значи да је његова интерна структура невидљива за корисника тог објекта. Кориснику је омогућен приступ објекту само преко скупа порука дефинисаних за тај објекат. На пример, корисник објекта o1 не зна његове инстанцне променљиве, као ни начин имплементације његових метода, па тако нема могућност приступа податку датум рођења. За разлику од оваквог, "чистог" објектно оријентисаног модела, већи број објектних система гради се на објектном моделу који омогућује дефинисање "јавних" инстанцних променљивих, видљивих сваком кориснику објекта, које онда дају кориснику могућност како читања вредности, тако и промене вредности инстанцне променљиве. [1]

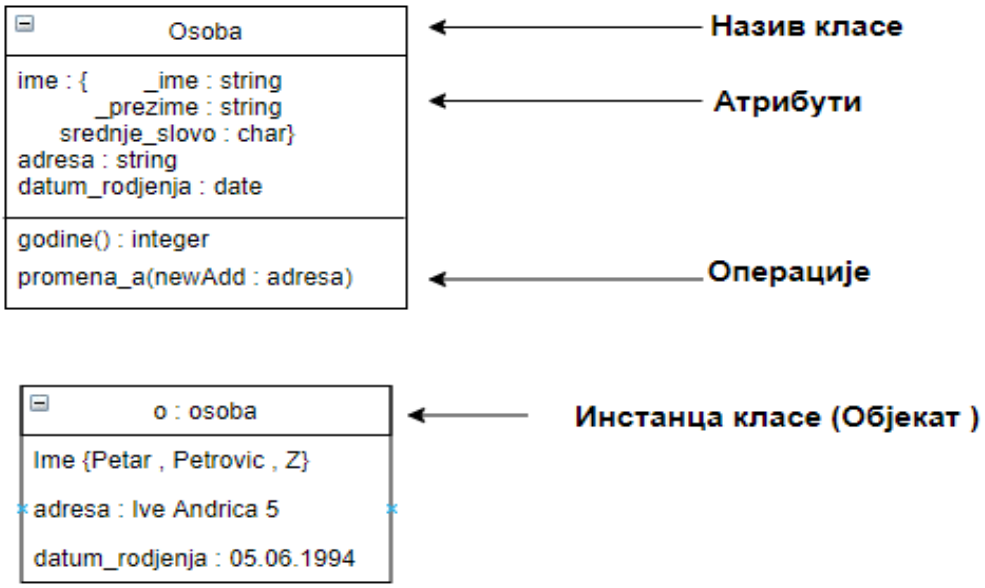
3.4 Класа

Да би настали објекти мора постојати нешто што их конструише,нешто што дефинише њихову стуктуру.Ово управо обављају класе, оне су заслужне за изглед структуре сваког објекта појединачно,тј тачније то за нас ради посебан објекат у класи који је задужен за изглед структуре инстанци, примерка класе које прави.Такође поред вредности може садржати и операције и функције које над тим објектима могу извршавати.Свака класа садржи методу „new" за креирање примерака класе, ова метода се примењује на објекат класе.Сада ћемо направити једну класу и инстанцу класе како би смо практично објаснили ту вези између класа и објеката.Пре тога једна илустрација која показује природу односа класе и објекта. [2]



Слика 8. Илустровани приказ класе и објеката на примеру аутомобила

Следи пример класа и објекта и њихове организованости.



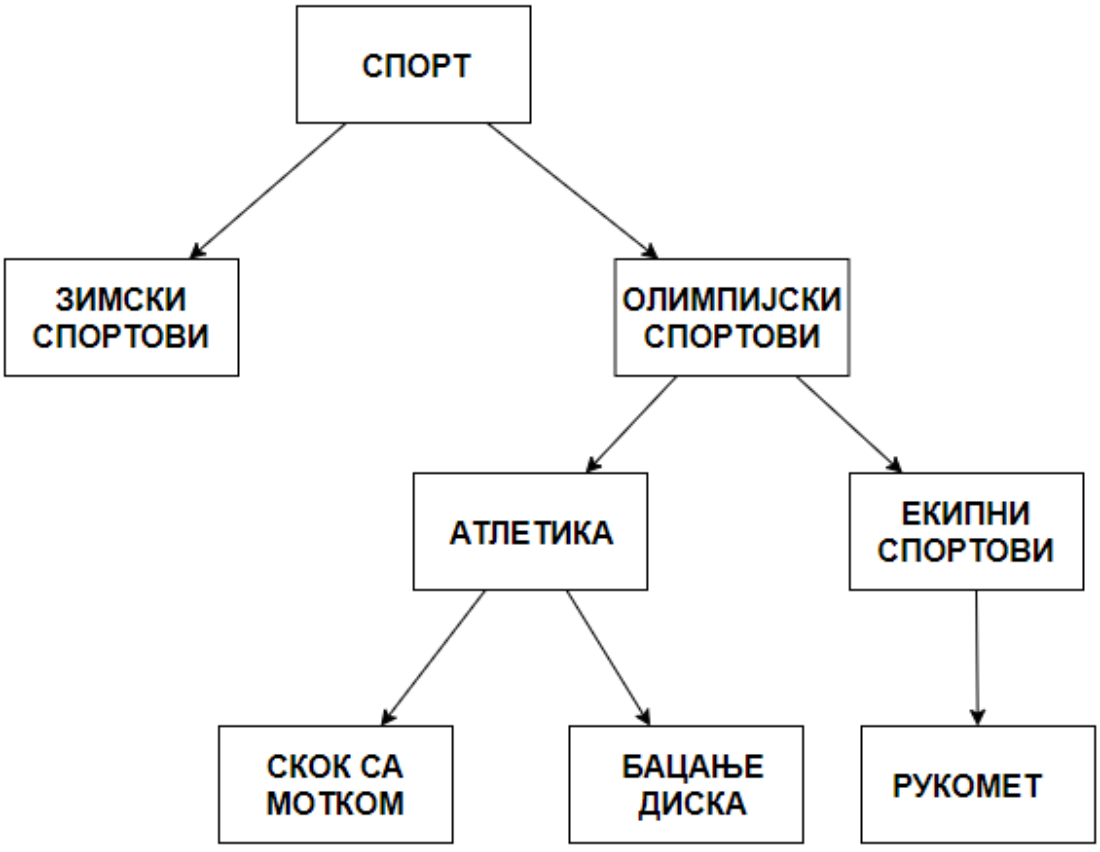
Слика 10 Класа и инстанца класе (објекат)

Са слике 10 се види класа која има назив „Osoba“, на слици се налази у заглављу, док су испод заглавља а у централном блоку смештени атрибути.Овде се налазе вредности које ће при инстанцирању сваки објекат добити.Те вредности су име који је сложени атрибут (име , презиме ,средње слово), адреса и ,датум рођења.Ове вредности управо чине структуру сваког објекта примерка класе „Osoba“.
На самом дну у последњем блоку на слици 10 се налазе операције и методе које можемо вршити над објектима,а које су дефинисане у оквиру класе „Osoba“.Те операције су „godine()“ (нпр. операција може рецимо рачунати број година на основу датума рођења са којим располаже) и „promena_a“ (операција која мења адресу особе). Свака инстанца класе „Osoba“. Има структуру дефинисану у класи, али што се тиче операција оне се примењују над објектима али не и на свим инстанцама.То су методе које можемо користити над објектима по потреби нису примарне тј. не улазе у саму структуру објекта. [2]

Испод класе „Osoba“ налази се пример инстанце (објекта) о којој смо говорили.Као што се може видети са слике 7. објекат има структуру добијену из класе.Ту се налази име презиме и средње слово са вредностима коју тај објекат има у виду стринга („Petar Petrovic Z“).Адреса и датум рођења су такође у саставу објекта и имају своју вредност.[1]

3.5 Наслеђивање

Када се говори о објектним моделима база података једно од битинијих могућности јесте наслеђивање.Као што сама реч каже да би се нешто наследило морамо имати неки већ дефинисани концепт, у нашем случају је то класа од које наслеђујемо све инстанце променљиве и методе од те класе.Ово значи да је сваки примерак подкласе уједно и примерак надкласе (класа од које смо наследили структуру и методе).Ово својство доприноси развоју већ постојећег система.Под класа може и обично има своје методе и инстанце.Методе које се наслеђују од надкласе могу бити редефинисане.Могућност да се метода примењује у различитим класама и типовима зове се полиморфизам.Не постоји ограничен број подкласа. Постоје системи где је могуће наследити само једну класу тј. једна класа највише може имати једну надкласу и овакав концепт се назива једноструким наслеђивањем. Супротно томе вишеструко наслеђивање подразумева да класа може имати већи број надкласа.
На следећој слици је приказан однос класа и подкласа класе „СПОРТ“.

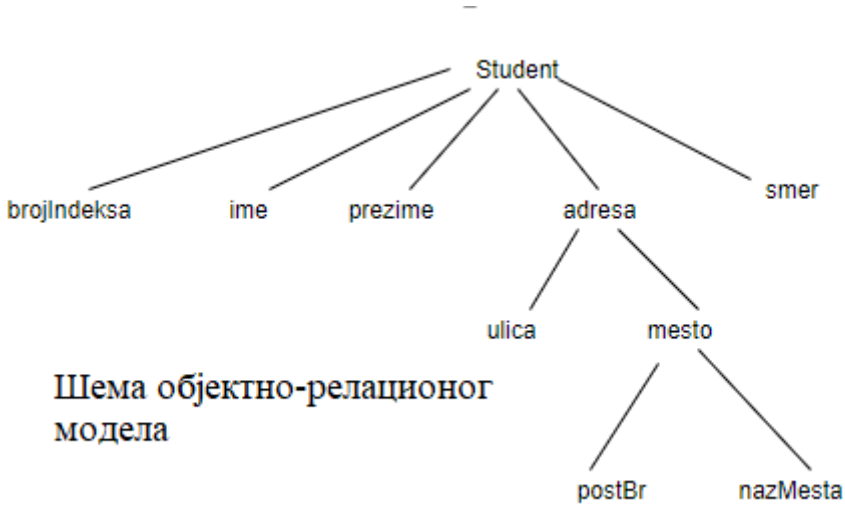


Слика 11. Вертикална хијерархија класе СПОРТ

Као што се може видети са слике, класа „СПОРТ“ је над класа класама „ЗИМСКИ СПОРТОВИ“ и „ОЛИМПИЈСКИ СПОРТОВИ“.Даље класа „ОЛИМПИСКИ СПОРТОВИ“ постаје надкласа класама „АТЛЕТИКА“ и „ЕКИПНИ СПОРТОВИ“.На овом једноставном примеру може се видети предност објектног система са наслеђивањем јер није потребно за сваки спорт правити нове делове нове надкласе већ можемо наставити са надоградњом модела.Пример ако би настао нови спорт различит од постојећих лако га је додати и наследити оно што је заједничко за све спортове.

4. Објектно-релациони модел

Када је било речи о објектном моделу говорило се да су се у почетку уводиле новине у релационим базама те је тек касније настао потпуно нови концепт.Објектно релациони модел је управо унапређени релациони модел.Објектно-релациони модел користи релациони модел, док га SQL тумачи.За разлику од класичног релационог модела којег познајемо и који смо описали у првим поглављима објектно релациони модел је отворен за нове начине и функције.Нове врсте података и функције се могу имплементирати коришћењем језика опште намене као што су „C“и „Java“.Оно што желимо да истакнемо и што представља једну од битнијих разлика јесте да програмерима омогућава уградњу нове класе, објекта података у релационом моделу. [4]



Шема објектно-релационог модела

Шема ORDBMS (Object-Relations-Database-Menager-System) има додатне функције у односу на RDBMS (Relations-Database-Menager-System).Неколико структура објектно-орјентисаног модела је део карактерисика ORDBMS као што су полиморфизам и наслеђивање које се разјашњавало у поглављу које се бавило објекто-орјентисаним системом.

Сваки приступ подацима у ORDBMS – у обрађује се декларативним SQL изјавама.Битна проширења релацијског модела која ћемо истаћи су:

- уведена објектна оријентација и конструкције које омогућују руковање са новим типовима података
- дозвољава да атрибути имају сложене вредности, укључујући и угњеждене релације (нарушена 1NF)
- већи избор података што проширује област примене

brojIndeksa	ime	prezime	adresa			smer
			ulica	mesto		
				postBr	nazMesta	
17/2017	Petar	Petrovic	Ive Andrica 5	2500	Beograd	Informatika u inz.
17/2017	Jovan	Krsic	Zmaj jovina 24	5502	Novi Sad	Proizvodno masinstvo

Табела 1. Табела објектно релационе базе

У поглављу које се бави релационим моделом видели смо да се подаци налазе у табелама.У објектно-релационим базама се подаци такође налазе у табели налик на релацијски модел.
На примеру из табеле 1 атрибут адреса је сложени атрибут.По правилима, односно нормалним формама релационог модела ова шема се не налази у 1NF али објектно-релационе базе података дозвољавају нарушавање овог правила које говори о томе да свака торка мора бити атомичне вредности.То није случај када је у питању адреса јер се види да адреса има структуру објекта чија се вредност састоји из вредности улица и место.Даље се може приметити да се и место састоји из postBr и nazMesta што чини и овај атрибут сложеним.[4]

Оно што се мора нагласити да не постоји јединствени модел објектно-релацијских база.Они се међусобно разликују.Разлике се могу уочити у степену проширења релационог модела.Да би се разумели о каквим додацима релацијама се говори истаћиће се нека од могућих проширења :

- апстрактни типови података (објектни типови, кориснички-дефинисани типови, ...)
- идентификатори објекта и референце
- методе за објектне типове, учаурење
- кориснички дефинсан „CAST“
- објектне таблице
- наслеђивање типова и таблица
- угњеждене релације (сложени атрибути, колекције)

Због могућих проширења а и разлика између самих ОР модела јављају се и различити системи за управљање базама података.Тако да концепти ОР модела нису комплетно имплементирани нити у једном СУБП (Систему за управљање базама података). Па тако, различити СУБП користе различите приступе што је једна од мана овог модела. SQL 1999 у себи подразумева већину објектно-релационих концепата као што су:

- Конструктори типова за дефинисање сложених објеката
- Индентитет објеката
- Учауривање
- Наслеђивање

Ово су уједно и концепти који се налазе у већини ОР концепата па је и логично подржати неким системом ове надоградње релационих база. [4]

4.1 Подела типова података

Када је реч о поделама типова података у објектно-релационим моделу, њих можемо сврстати у неке три групе :

1. Унапред дефинисани типови података (predefined types)
2. Изграђени типови података (constructed types)
3. Корисничко дефинисани типов

4.2 Изграђени типови података

Када говоримо о овој врсти података говоримо управо о проширењу класичног релационог модела.Када смо објашњавали у поглављу које се бавило објектним моделом ми смо говорили да је могуће направити конструктор који ће моћи да изгради објекте који имају сложен скуп података.Овде је то идентично али пренесено у релациони модел па тако је могуће уз помоћ конструктора направити такве типове података да свака вредност буде направљена од једне или више вредности које припадају истим колонама или могу припадати различитим типовима података.Овакви комбиновани типови података се називају композитне вредности.Вратићемо се на пример и табелу објектно релационог модела и издвојити једну композитну вредност. [4]

adresa		
ulica	mesto	
	postBr	nazMesta
Ive Andrica 5	2500	Beograd
Zmaj jovina 24	5502	Novi Sad

Слика 12. Композитна вредност

Имамо шему где се у редовима налазе различити типови података. Адреса је дакле композитна вредност и састављена је од вредности места и улице.
Треба рећи да изграђене вредности могу бити и атомичне.Оне се састоје од једног типа података чија вредност показије на локацију на којој је сачувана вредност референтног типа.Могу показивати само н-торке табела изграђеним на структурном типу, то значи да могу показивати на вредност типа улица која се налази у структури адресе али не и места јер је она композитна вредност слика 12.

4.3 Кориснички дефинисани типови података

Кориснички дефинисани типови (user-defined types) су подаци који су дефинисани и именовани од стране корисника.Називају се и абстрактним типовима података (abstract data types).Постоје две групе :

1. Distinct тип

Кориснички је дефинисан .Темељи се на вредностма које су атомичне. Над овим типом података могуће је дефинисати методе,али није могућа хијерархија.

```
CREATE TYPE godineT AS INTEGER FINAL;
CREATE TYPE tezinaT AS INTEGER FINAL;
CREATE TABLE osoba (
  sifOsoba INTEGER
  prezime VARCHAR(25),
  godineOsoba godineT
  tezinaOsoba tezinaT
  PRIMARY KEY sifOsoba);
```

Слика 13. Кориснички дефинисане вредности

Могуће је упоређивати вредности истог Distinct типа, међутим није могуће упоређивати вредности Distinct типа и атомичног типа на којем је темељен те различит Distinct типова темељених на истом атомичном типу. [4]

```
SELECT sifraOsoba FROM osoba                /* ERROR */
WHERE (godineOsoba * 2) < tezinaOsoba;   /* ERROR */
```

Слика 14 Пример упоређивања различитих типова података

Упоређивање вредности Distinct типа и атомарног типа на којем је темељен дозвољено је уз коришћење функције CAST.

```
SELECT sifpaOsoba FROM osoba
WHERE CAST(godineOsoba AS INTEGER) * 2 < CAST(tezinaOsoba AS INTEGER);
```

Слика 15 Пример упоређивања типова уз функцију CAST

4.4 Структурни тип

Структурни тип именовани кориснички дефинисани тип података.Произвољне је и сложене структуре.Атрибут припада именованој компоненти структурног типа.Ево једног примера структурног типа.

```
CREATE TYPE mesto AS (
        Postanski_Br INTEGER,
        naziv_Mesta VARCHAR(40))
NOT FINAL;    /*Not Final- dozvoljeno je kreirati podtip tipa */
```

Слика 16. Структурни тип „mesto“

Инстанца структурног типа података је вредност,али има неке карактеристике објекта.Та вредност структурног типа се састоји од одређеног броја вредности атрибута. [3]

Структурни типови се могу користити као:

- тип атрибута другог структурног типа
- тип параметра функција, метода и процедура
- тип SQL променљиве
- тип атрибута

```
CREATE TYPE mestoT AS (
postanski_Br INTEGER,
naziv_Mesta VARCHAR(40)
) NOT FINAL;

CREATE TYPE adresaT AS (
ulica VARCHAR(50),
mesto mestoT
) NOT FINAL;

CREATE TABLE osoba (
sifOsoba INTEGER ,
ime VARCHAR(25),
prezime VARCHAR(25),
adresa adresaT );
```

Слика 17. Креирање табеле са атрибутима који су структурног

5. Примери релационог,објектног и релационо-објектног модела

У овом поглављу се говори о разликама између релационог , објектног и објектно-релационог модела.Да би се разлике уочиле, узећемо пример студената за потребе факултета и студентске службе.Нећемо преводити комплетан информациони систем студентске службе,већ ће се узети релација студент и разматрати.Пројектоваћемо пример на сва три модела и уочити могуће недостатке једног у односу на друге системе за сваки модел понаособ.Моделе које разматрамо су :

1. Релациони модел
2. Објектни модел
3. Објектно-релациони модел

5.1 Релациони модел- пример

Прво што ћемо показати јесте како правимо табеле у релационом моделу помоћу SQL команди.Помоћу овог језика за манипулацију са базама података врло лако можемо направити ову релацију.то изгледа овако :

```
CREATE TABLE Student (
    Br_indeksa varchar(255),
    Ime varchar(255),
    Prezime varchar(255),
    NazivSmera varchar(255),
    godine_studija int,
    PRIMARY KEY (Br_indeksa),
);
```

Слика 18. Креирање табеле SQL командом

Број индекса ће бити примарни кључ док ћемо адресу ставити као страни кључ.

Br_indeksa	Ime	Prezime	NazivSmera	godine_studija
40/2015	Petar	Petrovic	Inf. u inženjerstvu	3

Табела 2. Изглед табеле након креирања

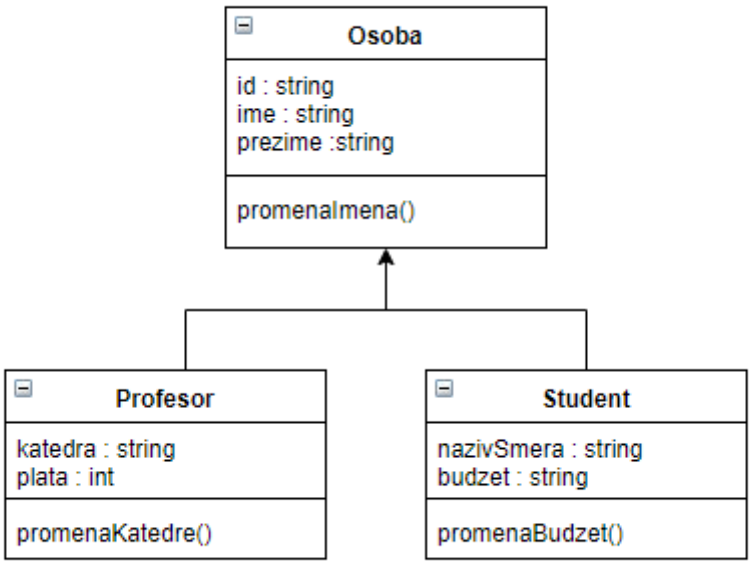
Ако желимо да убацимо адресу студента неопходно је да имамо засебну табелу, јер све вредности морају бити атомичне.На следећој слици је приказано како креирати табелу „Adresa“ и дефинисати страни и примарни кључ.

```
CREATE TABLE Adresa (
    id_A int NOT NULL,
    Br_indeksa int NOT NULL,
    Drzava varchar(255),
    ulica varchar(255),
    broj int,
    PRIMARY KEY (id_A),
    FOREIGN KEY (Br_indeksa) REFERENCES Student(Br_indeksa)
);
```

Слика 19 Креирање адресе

5.2 Објектни модел- пример

У овом моделу подаци се не смештају у табелама.Они су организовани да представљају једну колекцију објеката који су међусобно повезани.Сада ћемо показати како би изгледала шема овог модела.



Слика 20 Шема објектног модела

Дакле имамо класу особа.Ова класа у себи садржи елементе име, презиме, и метуду којом можемо променити ове вредности.Пошто свака особа има име и презиме инстанце објеката ће имати управо ове карактеристике.Објекат студент као објекат има име, презиме и остале своје вредности.Предност оваквог модела је у томе што када би смо хтели да додамо професоре у информациони систем они би били такође инстанце ове класе али имали би своја обележја која их карактеришу.На дну објеката (инстанци) се налазе методе које можемо вршити над објектима тог типа. Ако студент не оствари буџетско место користимо методу „promenaBudzeta“ и објекат добија нову вредност променљиве „budzet“.

5.3 Објектно-релациони модел – пример

Објектно-релациони модел се базира на релационом моделу.Вредности су такође смештене у релацијама ,а њима се управља SQL-ом.
Пошто је начин на који се креирају табеле идентичан нећемо се бавити тиме, већ ћемо показати како се у релационим базама могу дефинисати у табели сложени атрибути релације.то вршимо на следећи начин :

```
CREATE TABLE Student (
        Br_Indeksa INTEGER,
        ime VARCHAR(25),
        prezime VARCHAR(25),
        smer VARCHAR(25),
                adresa ROW ( ulica VARCHAR(50),
                             mesto ROW ( Postanski_Br INTEGER,
                                         naziv_Mesta VARCHAR(40))
                           )
);
```

Слика 21 Креирање сложеног атрибута релације „Student“

brojIndeksa	ime	prezime	adresa			smer
			ulica	mesto		
				Postanski_Br	Naziv_Mesta	
17/2017	Petar	Petrovic	Ive Andrica 5	2500	Beograd	Informatika u inz.

Табела 3. Изглед табеле након креирања

Када је било речи о релационом моделу рекли смо да није могуће убацити сложени атрибут у табелу тј. вредности морају бити атомичне, Па смо морали да правимо засебну табелу у којој смо дефинисали обележја адресе.Код објектно-релационог модела могуће је дефинисати сложени атрибут.Ово је само једна од разлика ова два модела.Објектно-релациони модел је овим нарушио 1NF (Прву нормалу форму) која глас : „Све вредности табеле морају бити атомичне (просте) и не смеју се понављати“.то заправоо значи да се атрибут не може даље раставити.С'обзиром да се адреса може раставити на улицу, место, а место на поштански број и назив места,адреса нарушава ово правило.Ово је једна од разлика у наредном поглављу ћемо детаљније анализирати предности и мане ова три модела.

6.Предности и мане модела база података

Приказ сличности и разлика између релационог моделирања података и објектно оријентисаних модела података је од великог значаја како за дизајнере базе података тако и за кориснике.Знајући да је добро упознат са релационим моделом и напомењући сличности и разлике између два приступа моделирању података (објектном и релационом), дизајнери ће моћи да то претворе у рачун и искористе већ стечено искуство као важну основу за учење методологије дизајнирања објектно оријентисаних база података. У време када дизајнери знају сличности и разлике између ова два приступа имају могућност претварања релационог модела у објектно оријентисан модел и обрнуто.Овакво знање не само да доприноси брзини и ефикасности, него и смањује трошкове и цену одређеног производа, јер су то повезане ствари.У овом раду у наставку ћемо упоређивати RDBMS, OODBMS и ORDBMS. [5]

6.1 Поређење РДБМС са ООДБМС-ом

Суштинска разлика између ова два типа моделирања података је представљена енкапсулацијом у објекту , стања и понашања са објектно оријентисаним моделом, док је са релационим моделом доказано само стање.

Релацијска база података састоји се од релација, који су скупови н-торки, док је објектно-оријентисана база података састављена од класа, и скупова класа.
Стога, релацијска база података ће садржати релацију која се зове СТУДЕНТ, а н-торка садрже информације о том ученику, док објектна база података садржи класу СТУДЕНТ, са објектом који садржи информацију о том ученику.
Треба напоменути да постоји могућност претварања објектног модела у релациони модел.
У таквој ситуацији свака класа одговара релацији, атрибути одређене класе ће постати атрибути који одговара тој релацији, сваки објекта у класи ће имати одговарајуће н-торке у релацији.
Док су у релационој бази података, компоненте н-торки примитивни типови (стрингови, цели бројеви,итд.), У објектно оријентираној бази података компоненте могу бити сложени типови (скупови, итд.). [5]

Табела 1 приказује поређење главних концепата који се користе у објектном и релационом моделирању података.

Објектно орјентисани модел	Релациони модел	Разлике
Објекат	Ентитет	Објект одређује понашање
Класа објеката	Типови Ентитета	Класа објеката укључује заједничко понашање објеката у тој класи
Хијерархија класа	Шема базе података	Хијерархија класа омогућава наслеђивање док релациона шема има страни кључ
Инстанца класе	Ентитет или н-торка	Инстанца може имати ограничавајући карактер
Атрибут	Атрибут	Нема разлика
Везе	Везе	Нема разлике.Али са ООДБМС, наслеђивање укључује и стање и понашање
Поруке / Интерфејс	Нема их	
Енкапсулација	Не постоји	
Индентификатор објеката	Примарни кључ	У релационом моделу ако је примарни кључ није индентификован,систем аутомацки индентификује кључ
Наслеђивање	Не постоји	

Табела 4.Табеларни приказ разлика објектног и релационог модела

6.2 Поређење ORDBMS са ORDBMS -ом

Што се тиче поређења објектно-релационог и релационог модела,чини се да они мају доста заједничког, међутим када говоримо о разликама,предностима и манама разматрађемо следеће особине и критеријуме који један односно други задовољавају.Погледајмо зато табелу 2 која ће нам одговорити на дилеме.

Критеријуми	ORDBMS	ORDBMS
Дефинисани стандард	SQL	SQL3
Подржаност објектно-орјентисаних карактеристика	Не подржава објектне карактеристике.Теско је превестии објекат у базу података са овим моделом	Ограничено подржава,углавном због нових типова података
Коришћење са аспекта корисника	Једноставан је за коришћење	Једноставан је за коришћење,осим неких екстензија
Подржаност за комплексне податке	Не подржава абстракне типове података	Подржава абстрактне типове података и подржава слошене везе
Перформансе	Веома добре перформансе	Веома добре перформансе
Настанак модела	Релативно стар систем	Још увек у фази развоја
Употреба SQL-а	Широка примена SQL	SQL3 се развија са објектно-орјентисаним карактеристикама уграђеним у њега
Предности	Зависган од SQL-а,релативно једноставна оптимизациј аупита па отуда и одличне перформансе	Способност за подршку комплексних апликација са комплексним захтевима
Недостаци	Неможе да подржи велике апликације	Лоше перформансе када су у питању апликације на „web“-у

Табела 5. Табела са карактеристикама модела

Као што смо могли из табеле да прочитамо сваки од модела има своје предности и недостатке.
Програмери који пројектују базе података морају имати у виду неке од наведених предности и мана како би информациони систем био ваљано испројектован.Када је мишљење људи који раде на овим моделима у питању, мишљења су подељена.Како год ови модели ће наставити са побољшавањем перформанси. [5]

7. ЗАКЉУЧАК

Базе података као област у „IT“ свету заузима посебно и важно место.Величина и број података је у сталном расту, ова чињеница захтева нова иновативна решења складиштења и манипулисања тим подацима.Објектно-релационе базе то свакако јесу.Њен претходник тј. релациони модел није имао ту могућност да подржи апликације сложене структуре, с'тога закључујем да ће се објектно-релациони модел развијати у будућности.Као и сви модели које смо описали и овај модел има доста мана.Оно што бих издвојио посебно јесте не постојање неког стандарда.Стандардизација у релационом моделу је спроведен, односно модел је јасно дефинисан док се за објектно-релациони модел то не може рећи.

ЛИТЕРАТУРА

[1] Гордана Павловић-Лазетић, „*Увод у релационе базе података“* Математички факултет, Београд.

[2] 2. R.G.G. Cattell – ”Object Data Management“., Ed. Addison Wesley , 1994.

[3] Интернет страница : https://www.w3schools.com/sql/

[4] Jim Melton: Advanced SQL: 1999 - Understanding Object-Relational and Other Advanced Features, Morgan Kaufmann, 2002

[5] M. Stonebraker, D. Moore: Object-Relational DBMSs: The Next Great Wave, Morgan Kaufmann Publichers, 1996

[6] https://www.draw.io/