БАЗИ НА ПОДАТОЦИ

Contents

Физичка организација на податоците

Основни дефиниции:

База на податоци – збирка на поврзани податоци
Податок – Познат факт кој може да се зачува и има имплицитно значење
Мал свет – Дел од светот за кој се чуваат податоците
Систем за управување со бази – Софтверски пакет/систем преку кој се создава и одржува електронска БНП
Систен на БНП – СУБП софтвер + податоци

Базите најчесто се чуваат во секундарна меморија, бидејќи нуди голем капацитет за мала цена.

Една трака може да е поделена на блокови наречени сектори (512 до 4096 бајти)
Адреса на блокот се состои од :
Број на цилиндар, број на трака, број на блок
Читање и запишување е скапо.

Записите содржат **полиња** кои имаат вредности од одреден тип.
Полињата може да бидат со фиксна или променлива должина.

Запишување по блокови – Blocking :
Сместување записи во еден блок (Записи по блок – blocking factor (BFR))
Може да има неискористен простор доколку неможе да се запише цел запис во блокот.

Датотека е секвенца од записи, која има file descriptor што ја опишува.
Записите во датотеката може да имаат фиксна или променлива должина.

Heap (pile) – нови записи се стават на крај,
Брзо пишување : споро читање

Подредени датотеки – записите се зачувуваат според клуч
Спор запис : брзо барање
Кај овие често има Overflow датотека во која се стават новите податоци пред да се стават во главната.

Hash датотеки
Датотеката се дели во кофи, и во која кофа ќе биде ставен еден блок се одредува со Hash функција.
Проблем: голем простор за каталог на кофи, и колизии.
Колизиите се решаваат со листа на прелеани записи.

-Отворено адресирање, Каде се става на првата слободна позиција.
-Нанижување, се става покажувач кон просторот за преливање.
-Повеќекратно расфрлање, Програмат користи нова хеш функција.
Фиксен број на кофи, лошо при намалување и зголемување на датотека, неефикасност во редоследен пристап до клуч /

Индексни структури:

-Еднонивовски
  Примарни индекси

Кластерирачки -=-
Секундарни -=-
-Повеќенивовски
-Динамички повеќенивовски со користење на дрва

Еднонивовски индекс е помошна датотека преку која се извршува поефикасно пребарување на запис.
Се спрецифицира над едно поле. <FIELD VALUE, POINTER>
Индексот се нарекува притапна падека до полето.
Индексната датотека зафаќа мал простор.
Се наоѓа покажувачот преку бинарно пребарување.

Густ индекс содржи индекс за секоја вредност од клучот
Реткиот индекс има индексни вредности само за некои од пребаруваните вредности

Примарен индекс
Се дефинира над подредена датотека (според клучно поле).
Вклучува еден индексен запис за секој блок

Кластерирачки индекс:
Се дефинира над подредена датотека,но сортирањето е над **не клучно** поле.
влучува индекс за секоја единствена вредност на полето по кое се сортира.
Внес и бришење се комплицирани

Секундарен индекс:
Секундарен пристап до датотеката до која постој примарен индекс
може да се изгради врз кандитат клуч или не клучно поле.
Секундарниот индекс се состои од поле од ист тип како полето врз кое се врши индексирање и покажувач кон блок или запис.
Овој индекс е густ.

Повеќенивовски индекси.
Бидејќи еднонивовскиот индекс е подредена датотека сама по себе, може да се создаваат индекси врз него.
Овој процес може да се повторува повеќе пати.

Ваквите индекси имаат форма на пребарувачко дрво.
Внесување и бришење е сложен проблем
Но брзо пребарување.

Решавање на проблемот на внесување е со оставање на простор на крајот на секој блок за нови записи.
Во B и B+ дрвата секој јазел е блок од дискот. Секој јазел е или пола или целосно полн.

Ако јазелот е полн се дели на 2.
ако јазелот е помалце од пола полн, се соединува.

В дрвото покажувачи кон податочни записи постојат во секое ниво
B+ Пок. Кон податочни записи само јак листови. Во B+ дрво може да има повеќе внесови

Ентитет релација (ЕР)

Функционалност на СУБП:
-Дефинирање на БНП
-Конструкција или полнење на иницијалните содржини од БНП
-Манипулација на БНП
-Процесирање и делење

Други функционалности : Заштита од неавторизиран пристап, презентација и визуелизација на податоци, Одржување на базата за време на користење на апликација.

Дизајн на БП : Дизајн на база на податоци + дизајн на апликација (СЕ)

Шема на БНП дава опис на БНП.
Структура, податочни типови, ограничувања.

Се состои од:
Ентитети – предмети или нешта од малиот свет кои се претставени во базата
Атрибути – Особини со кои се опишува ентитетот.

Атрибутите можат да бидат **едноставни** (секој ентитет има една вредност за атрибутот) или **сложени(**атрибутот може да биде составен од неколку состојќи пр. адреса**)**.

Повеќе-вредносен атрибут , пр. Боја на кола

Ентитетите од ист вид се организираат во еден **тип на ентитет.**
Одреден атрибут на некој ентитет, за кои сите ентитети имаат различна вредност се вика **клучен атрибут.**

Еден ентитет може да има повеќе од еден клуч. Секој клуч се подвлекува.
Ентитет во ЕР се прикажува со правоаголник.
Атрибут се прикажува со елипса.
Секој тип на ентитет ќе има **множество на ентитети снимени во базата.**

Помеѓу ентитети може да постој **релација.**
Релацијата е однос, и се однесува на различни ентитети со посебно значење.
Релации од ист тип се групирани во **тип на релација(**Опис на релацијата, назив и тип на ентитет кои учествуваат во неа, и ограничувања кои ги има**).**

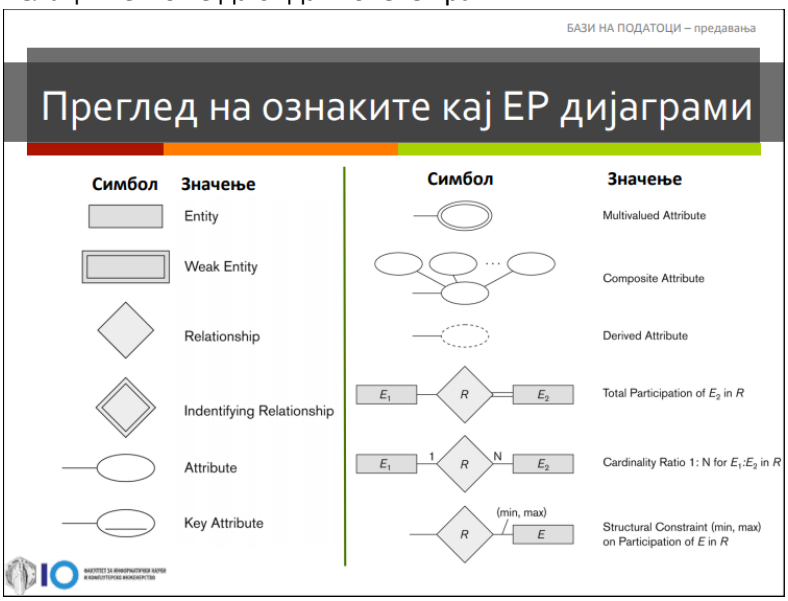
Кратноста на една релација е бројот на типови на ентитети што учествуваат во неа.

Множество релации – тековно множество примероци претставени во базата.
Релацијата се прикажува со ромб.
Во една релација може да учествува ист тип на ентитет повеќе од еднаш.

Кај релациите имаме ограничувања на **кардиналност (**1-н, 1-1 , н-1 , н – м **) и учество (**Целосно или делумно**).**
Кај типови на релации може да има и атрибут, и се употребуваат најчесто кај МН релација.
Кај 1-Н релациите атрибутите се пренесуваат кај типот на ентитет кај Н страната.

Ентитет што нема клучен атрибут, се нарекува **слаб ентитет** и мора да припаѓа на идентификувачка релација со ентитет сопственик.

Релациите може да бидат повеќе кратни.



Пер е подобрување на ЕР.
Има наследување(пр. Вработен = секретар).
Поткласи или подвидови.

Класа наткласа е подмножество ⊂
Ако има повеќе поткласи се користи О за спојување со наткласа.
Секоја поткласа може да има свои атрибути.
Секој поткласа може да влегува во специфични релации.
Генерализација е издвојување на заеднички особини на неколку класи во една нат класа.
Специјализација е дефинирање на множество пот класи од една класа.

Ограничување за одвоеност
Одвоени (еден ентитет припаѓа само на една поткласа **d**) или преклопувачки (еден ентитет може да припаѓа на повеќе пот класи **О**)

Ограничување за целосност

Потполно (секој ентитет на наткласата мора да е член на поткласа) или Делумно (може ентитет да не припаѓа на поткласа)

Во некои случаи треба да се моделира повеќе од една надкласа подкласа врска.
Ваква подкласа се нарекува **Категорија или union тип(**се означува со U**).**
Подмножество од пресек на нејзините надкласи. Ентитетите во делената подкласа мора да постојат во сите нејзини родителски надкласи.

Формални дефиниции:

КЛАСА С:
Ентитетен тип со соодветно множество ентитети (ентитетен тип, подкласа, надкласа, категорија)

Подкласа S:
Ги наследува сите атрибути и релации од класата
Ентитетно множество е подмножество на друга класа С.

Релациски модел на бази на податоци

Неформални изрази		Формални изрази
Табела (Table)		Релација (Relation)
Заглавие на колона (Column Header)		Атрибут (Attribute)
Сите можни вредности на колоната (All possible Column Values)		Домен (Domain)
Редица (Row)		Торка (Tuple)
Дефиниција на табелата (Table Definition)		Релациска шема (Schema of a Relation)
Пополнета табела (Populated Table)		Состојба на релацијата (State of the Relation)

Релациски модел се основа на поимот **релација.**
Релацијата претставува табела со вредности.
Релацијата содржи множество редици.
Податоците од секој ред претставуваат факти кои припаѓаат на ентитет од реалниот свет.
Секоја колона има заглавие кое кажува значењето на податоците од таа колона.

R (A1, A2,…., An)
Ai припаѓа на некој домен Д односно Д = ДОМ(А)
бројот на атрибути во релационата шема го одредува **степенот на релацијата.**
Релацијата е дефинирана како **множество од торки.**
Атрибутите во торката може произволно да е појават.

Ограничувања:
-Ограничувања на клучевите (**Key constraint**)
-Ограничувања на ентитетен интегритет (**Entity integrity constraint**)
-Ограничувања на референцијален интегритет (**Referential integrity constraint**)

Имплицитно ограничување е ограничување на **доменот.** Секоја вредност од торката мора да биде од доменот на својот атрибут(или 0).

Натклуч (Superkey) на R:
Група на атрибути од Р, за која важи дека не постоат 2 торки со иста вредност на натклучот
Овој услов мора да биде исполнет за било која важечка состојба на релацијата.

Клуч(Key)
Е минимален клуч, односно неможе да се извади од него атрибут без да стане не уникатен.
Ако една релација има повеќе кандидат клучеви, тогаш еден од нив се бира како **примарен клуч.**
Примарните клучеви се подвлекуваат.
Се препорачува за примарен клуч да се бира најкраткиот кандидат клуч.

Шема на релациска БП:
Множество С на релациски шеми кои припаѓаат на иста база на податоци + множество интегритетни ограничувања.
С е името на целата шема на базата на податоци.
S = {R1, R2, R3 ....., Rn}
Интегритетните ограничувања се референцијален и ентитетен интегритет.

Ентитетен интегритет: Примарниот клуч за секоја релација во С несмее да има НУЛЛ вредности во било која торка од релацијата. Ако примарниот клуч е составен од неколку атрибути, несмее ниеден од нив да има вредност НУЛЛ.
Референцијален интегритет: Ограничување што вклучува **две** релации. Се користи за да се специфицира **релација (**однос, врска**)** помеѓу торките во двете релации.

Множество на атрибути ФК во релациона шема Р е **надворешен клуч на Р1 кој се поврзува со Р2 доколку:**
1.Атрибутите од ФК имаат ист домен како и адтрибутите од примарниот клуч од Р2. Се вели дека овие атрибути ја **референцираат или се упатуваат на** релацијата Р2.

2.Вредноста на ФК во торката т1 се појавува или како вредност на ПК за некоја торка од Р2, или е НУЛЛ.
т(ФК) = т2(ПК). Велиме дека торката т1 ја **референцира или се упатува на торката т2.**
Р1 е релација која референцира, а Р2 релација која е референцирана.
Ако се исполнети овие две тогаш важи ограничувањето на референцијалниот интегритет помеѓу Р1 и Р2.

Кога БП се менува, се појавува нова состојба.
Операции со кои може да се мења БП се
ВНЕСИ
ИЗБРИШИ
ПРОМЕНИ

Мапирачки алгоритам ЕР во релација:
1.Мапирање на регуларни ентитетни типови
2.Мапирање на слаби ентитетни типови
3.Мапирање на бинарни 1:1 типови на релации
4.Мапирање на 1:Н типови на релации
5.Мапирање на М:Н типови на релации
6.Мапирање на мултивредносни атрибути
7.Мапирање на Н кратни типови релации
(кај ПЕР)
8.Мапирање на специјализација или генерализација

Чекор 1. Мапирање на регуларни ентитетни типови
За секој јак ентитет во шемата, се креира релација Р која ги содржи сите прости типови од атрибути
Сложените типови се трансформираат во листа на прости атрибути.
Се одбира еден (или повеќе) од клучните атрибути да биде примарен клуч

Чекор 2. Мапирање на слаби ентитетни типови
За секој слаб ентитет во шемата се прави релација Р, која ги содржи сите негови прости атрибути.
Исто така во Р се става атрибут како надворешен клуч, кој е дел од примарниот клуч на идентификувачкиот ентитетен тип.
Примарниот клуч на оваа релација ќе биде комбинацијата на примарните клучеви од идентификувачкиот идентитет и слабите клучеви на релацијата(доколку ги има). (сите се означуваат со подвлекување).

Чекор 3. Мапирање на 1:1 типови релации
1.Пристап со надворешен клуч
Се одбира една од релациите и се вклучува примарниот клуч на другата како надворешен во неа.
Препорака е да става клучот во релацијата со целосно учество.
2.Пристап со соединета релација
Се спојуваат двата ентитети во една релација. Ова е возможно кога и двата се со целосно учество
3.Правење на лук ап табела, нова релација која ги содржи примарните клучеви на двете релации.

Чекор 4. Мапирање на 1:Н типови релации
Во релацијата на Н страната, се вметнува примарниот клуч од релацијата на 1 страната.
доколку имало атрибути во релацискиот тип се вгнездуваат во Н страната.

Чекор 5. Мапирање на М:Н типови релации:
За секоја ваква релација се креира нова релација К. Во неа се стават 2 надворешни клуча земени од ,двата ентитети кои влегуваат во релацијата. Се формира сложен клуч.Доколку постојат атрибути, тие се стават во оваа табела.

Чекор 6. Мапирање на Повеќе вредносни атрибути:
За секој повеќе вредносен атрибут се креира нова релација.
Во неа како надворешен клуч го ставаме ПК од ентитетот од кој произлегла.
Се прави сложен клуч од вредноста на атрибутот и НК.

Чекор 7. Мапирање на Н кратни типови релации
Се креира нова релација. Во неа како надворешни клучеви се стават ПК на сите ентитети во релацијата.
Ако постојат атрибути се стават овде.

Чекор 8. Мапирање на специјализација или генерализација
Постојат повеќе опции
А: Повеќекратни релации (наткласи поткласи)
Креирај релација за главната наткласа, која ги има сите атрибути и примарен клуч
Потоа креирај релација за секој поткласа, која ги има сите нејзини атрибути и примарен клуч од наткласата.

В: Повеќекратни релации (за релации кои се поткласи)
Креирај релација за секоја поткласа, која ги има сите атрибути од поткласниот ентитет И наткласниот ентитет. примарен клуч Е ПК на наткласата. Најдобра е за поткласи кои се целосни (мора ентитетите да припаѓаат на поткласа).

С: Еднократни релации со еден тип атрибут
Креирај релација со сите атрибути од сите специјализации, примарен клуч ПК од наткласата, И атрибут ТИП кој означува од која специјализација е.

D: Еднократни релации со повеќе типови атрибути
Креирај релација со сите атрибути од сите специјализации, и по еден атрибут за секоја специјализација.
Овој атрибут ќе кажува дали припаѓа на таа специјализација

Преглед на операциите од релационата алгебра

Table 6.1
Operations of Relational Algebra

Operation	Purpose	Notation
SELECT	Selects all tuples that satisfy the selection condition from a relation R.	$\sigma_{<\text{selection condition}>}(R)$
PROJECT	Produces a new relation with only some of the attributes of R, and removes duplicate tuples.	$\pi_{<\text{attribute list}>}(R)$
THETA JOIN	Produces all combinations of tuples from R_1 and R_2 that satisfy the join condition.	$R_1 \bowtie_{<\text{join condition}>} R_2$
EQUIJOIN	Produces all the combinations of tuples from R_1 and R_2 that satisfy a join condition with only equality comparisons.	$R_1 \bowtie_{<\text{join condition}>} R_2$, OR $R_1 \bowtie_{(<\text{join attributes 1}>), (<\text{join attributes 2}>)} R_2$
NATURAL JOIN	Same as EQUIJOIN except that the join attributes of R_2 are not included in the resulting relation; if the join attributes have the same names, they do not have to be specified at all.	$R_1 *_{<\text{join condition}>} R_2$, OR $R_1 *_{(<\text{join attributes 1}>), (<\text{join attributes 2}>)} R_2$ OR $R_1 * R_2$
UNION	Produces a relation that includes all the tuples in R_1 or R_2 or both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cup R_2$
INTERSECTION	Produces a relation that includes all the tuples in both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cap R_2$
DIFFERENCE	Produces a relation that includes all the tuples in R_1 that are not in R_2; R_1 and R_2 must be union compatible.	$R_1 - R_2$
CARTESIAN PRODUCT	Produces a relation that has the attributes of R_1 and R_2 and includes as tuples all possible combinations of tuples from R_1 and R_2.	$R_1 \times R_2$
DIVISION	Produces a relation $R(X)$ that includes all tuples $t[X]$ in $R_1(Z)$ that appear in R_1 in combination with every tuple from $R_2(Y)$, where $Z = X \cup Y$.	$R_1(Z) \div R_2(Y)$

ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

R	
A	B
1	2
1	4
2	5

X

S	
B	C
2	6
2	4
3	5
4	8

=

Q			
A	B	B	C
1	2	2	6
1	2	2	4
1	2	3	5
1	2	4	8
1	4	2	6
1	4	2	4
1	4	3	5
1	4	4	8
2	5	2	6
2	5	2	4
2	5	3	5
2	5	4	8

A	B
a1	b1
a2	b1
a3	b1
a4	b1
a1	b2
a3	b2
a2	b3
a3	b3
a4	b3
a1	b4
a2	b4
a3	b4

A
a1
a2
a3

T

B
b1
b4

Постојат агрегатни функции кои се користат за правење на пресметки врз торките
COUNT , SUM , AVG , MIN, MAX …

$\mathcal{F}$COUNT SSN, AVERAGE Salary (EMPLOYEE)

Пред ф може да се стави групирачки атрибут

Outer join прави џоин ама ако нема вредност во втората торка тогаш става нул