Sveučilište u Zagrebu
Fakultet elektrotehnike i računarstva
Zavod za elektroničke sustave i obradu informacija

OBJEKTNO ORIJENTIRANE BAZE PODATAKA

Darijo Šplihal, 0036388804

Seminar: SPVP

2004/05

Zagreb, 04. 06. 2004

Sadržaj

1 Uvod

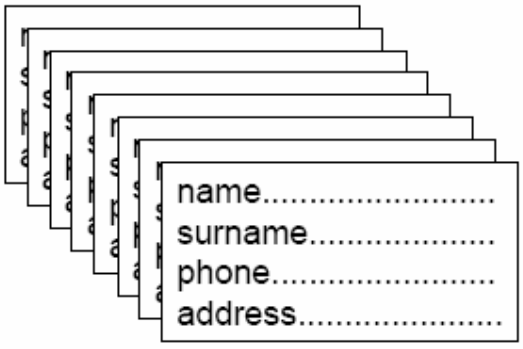
Ovaj članak prvenstveno je namijenjen upoznavanju objektno orijentiranih baza podataka, međutim kako bih što bolje objasnio objektno orijentirane baze potrebno je pripremiti teren. Što to točno znači? To znači da ću najprije objasniti neke temeljne osnove baza podataka, njihove početke, te što su ubiti baze podataka. Prvo ćemo se pozabaviti relacijskim bazama podataka koje će nas postupno dovesti do objektno orijentiranih baza podataka.

2 Relacijske baze podataka (RDB)

2.1 Što su baze podataka

Što je prva stvar što vam pada na pamet kad čujete naziv baza podataka?

Vjerojatno su nam svima poznati termini zapis i polje pa to nećemo objašnjavati. Na slici1 je nacrtana baza podatka kako bi si to prosječan čovjek koji ne zna ništa o bazama podataka predočio.

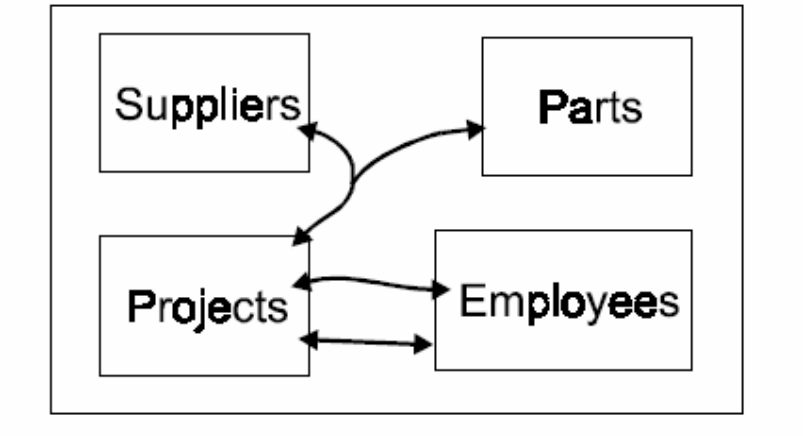


Slika 1. Baza podataka kako bi ju običan čovjek zamislio

Ako i vi ovako zamišljate bazu podataka, molim vas da izbrišete tu sliku iz glave. Razmišljanjem o bazi podataka kao o linearnom i homogenom zapisu jednako je grozno kao da mislite da je skateboard osnovno prijevozno sredstvo. Linearni zapis je jedan vrlo, vrlo ograničen zapis baze podataka.

Tipična baza podataka je zapravo jedan heterogeni skup međusobno povezanih informacija. Primjer baze podataka prikazan je na slici2. Ova slika predstavlja jedno poduzeće u kojem se nalaze *zaposlenici* koji rade na *projektima* (vidi strelice). Za

projekte potrebne su nam *komponente* koje se nabavljaju kod *nabavljača*.(trostruka strelica). Možete primijetiti da postoji još jedna dodatna strelica između zaposlenika i projekata a ona predstavlja drugačiji odnos od prve strelice. Ona predstavlja zaposlenike koji su ujedno i voditelji projekata. Za početak nećemo ići u detalje o tome kako su zapravo ove informacije i njihove veze pohranjene u bazama podataka. Ovim primjerom želio sam pokazati da baza podataka nije linearan zapis već jedan mnogo kompleksniji objekt.



Slika 2. Primjer poduzeća

Jedna od osnovnih svojstava baze podataka je mogućnost rješavanja kompleksnih i ugniježđenih upita kao što to pokazuje slika2

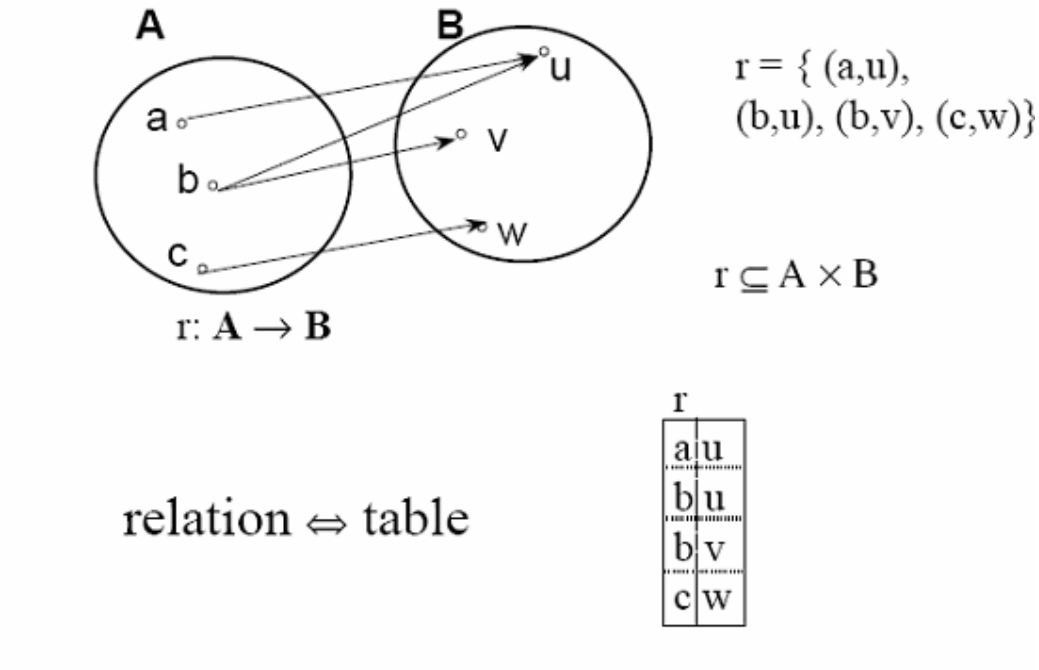
2.2 Što je relacijska baza podataka

Vjerojatno smo se, bar općenito, dogovorili o tome što je baza podataka, međutim nismo objasnili što znači pojam relacijski.

Formalna definicija je da relacijske baze podataka se bave podacima koji imaju relacijsku strukturu te sadrže jezik za obradu takvih podataka (DML) koji svoje temelje povlači iz relacijske matematike. Ovo je bila formalna definicija međutim ona trenutno nema baš previše smisla pa ćemo dati jedan jednostavniji odgovor.

Nazivaju se relacijske zato što pohranjuju svoje podatke u tablice koje su izomorfne matematičkim relacijama.

Sve dolazi iz matematičkog koncepta relacije. Slika3 ilustrira relaciju *r* između dva skupa, *A* i *B*. Relacija je podskup kartezijevog produkta dvaju setova. Ovdje je *r* podskup od $A \times B$ što je ubiti drugi način da se *r* opiše kao skup od 2-elementarnih n-torki kod kojih je prvi element iz A a drugi iz B.



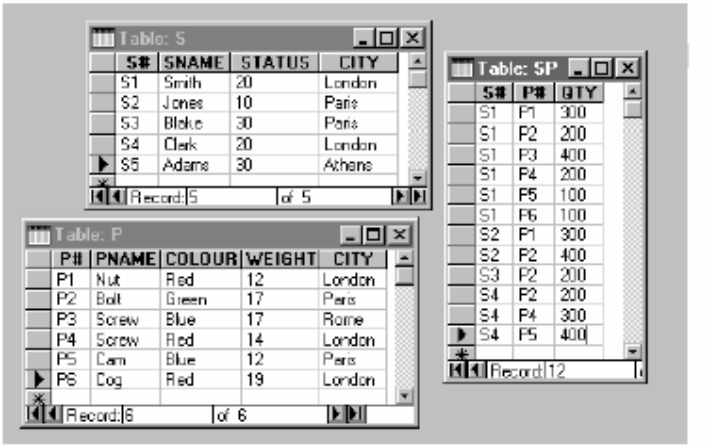
Slika 3. Matematički opis relacije

Za sada je dovoljno da shvatite da postoje dva zahtjeva: o strukturi podataka i o jeziku za obradu takvih podataka

Jezik za obradu podataka (DML) je programski jezik kojim se vrše određene operacije kojima se pretražuju i održavaju baze podataka. Da bi smo mogli vršiti ikakve radnje, nad primjerice podacima kao na slici2, potrebno je operacije koje mislimo obaviti najprije prevesti u DML oblik.

2.3 Osnovni pojmovi i koncepti

Osnovne pojmove i koncepte relacijske baze podataka lakše će biti objasniti na konkretnom primjeru.



Slika 4. Baza podataka

Dobavljači su navedeni u tablici S (na vrhu (*Supplier*)), a svaki red tablice predstavlja jednog dobavljača. Tablica je ubiti ekvivalentna homogenom zapisu sa početka, s tim da svaki redak predstavlja jedan zapis a svaki stupac polje. Međutim treba primijetiti da je cijela baza podataka sazdana od mnoštva takvih tablica. Tu se još nalaze tablica P (*Parts*) koja sadrži zapise o komponentama i SP tablica koja nam govori koje komponente i u kojoj količini se mogu naći kod kojeg dobavljača (SP tablica predstavlja jednu od onih strelica a tablice S i P su kvadratići sa slike2).

Svaki redak je lista od n vrijednosti (n je broj stupaca tablice) ili *n-torka* u matematičkom smislu.

Svaki stupac predstavlja polje zapisa i takav zapis naziva se *atribut*.

Atributi mogu poprimiti vrijednosti (npr. atribut «colour») samo iz određenog skupa, a takav skup dozvoljenih vrijednosti naziva se *domena*.

Ključ je atribut ili kombinacija atributa koja jedinstveno određuje redak tablice. Na primjer P# (redni broj komponente) je atribut koji jedinstveno određuje točno određenu komponentu, u smislu da ako dobijemo broj komponente znamo da postoji samo jedna komponenta (jedna n-torka) koja odgovara tom broju. Boja komponente ne određuje jedinstveno komponentu jer mogu postojati dvije različite komponente iste boje.

Treba primijetiti da svakoj n-torki u tablici SP odgovara n-torka u tablicama S i P ovisno o ključu. Postoje točno određena pravila koja definiraju vezu među tablicama ali nećemo ulaziti u detalje.

2.4 Zaključak

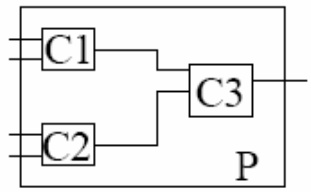
Osnovna ideja je jednostavna i elegantna: sve je prikazano u tablicama. DML je visokog stupnja, deklarativan i određen. Tablice su izomorfne matematičkim relacijama, što stavlja relacijski model baza na čvrste teorijske temelje što omogućuje razvoj teorema i dokaza istih. Relacijska algebra je zatvorena: algebarske operacije uzimaju relacije kao operande i kao rezultat vraćaju relacije, te time omogućuju ugnježđivanje. Međutim relacijske baze ne mogu riješiti sve probleme. Postoje određene podatkovne strukture koje se ne mogu staviti u tablice ili se nad takvim tablicama teško ili nikako ne mogu vršiti upiti.

3 Objektno orijentirane baze podataka (OODB)

3.1 Što ne valja sa relacijskim ?

Objektno orijentirane baze podataka pojavile su se sredinom 80-tih kao rješenje problema relacijskih baza koji su se pojavljivali u određenim klasama aplikacija.

Mi ćemo ove nedostatke prikazati na primjeru iz elektronike CAD.



Gate Instance	GT	GI	Parent
	2AND	C1	4AND
	2AND	C2	4AND
	2AND	C3	4AND
	4AND		

Pin Type	GT	PT	I/O
	2AND	A	I
	2AND	B	I
	2AND	C	O
	4AND	A	I
	4AND	B	I
	4AND	C	I
	4AND	D	I
	4AND	E	O

Gate Type	GT	Description
	2AND	C=A&B
	4AND	E=A&B&C&D

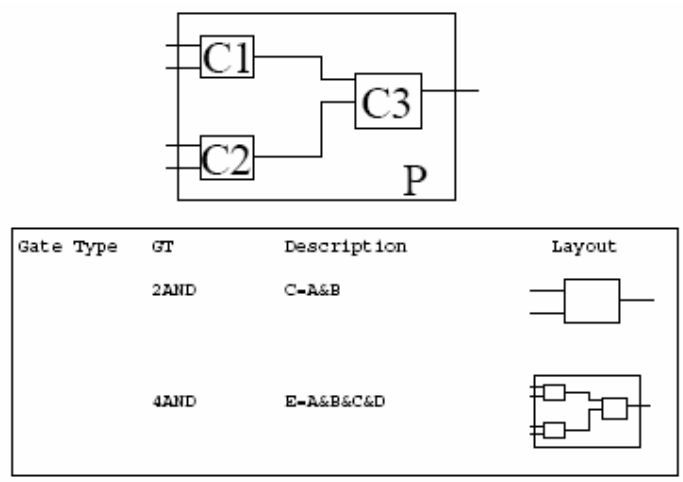
Wire Instance	WI	GT1	GI1	...
	W1	4AND	P	A
	W2	4AND	P	B
	W3	4AND	P	C
	W4	4AND	P	D
	W5	2AND	C1	C
	W6	2AND	C2	C
	W7	2AND	C3	C

Slika 5. Električni primjer

Imamo sistem koji se sastoji od međusobno povezanih logičkih vrata. Sastoji se od tri dvoulaznih I-vrata C1, C2 i C3 spojenih tako da čine četveroulazna I-vrata P. Kako bi smo ovo predočili u relacijskim bazama? Jedan od načina prikazan je u tablicama na

slici5. Napravili smo jednu tablicu koja opisuje tipove vrata te drugu koja sadrži instance vrata. Zatim tablicu sa tipovima pinova i jednu sa instancama žica.

Postoji nekoliko problema koji se javljaju pri ovakvom opisu. Rasipanjem osnovnih dijelova jednih jedinih vrata na više tablica je neprirodno i neintuitivno sa stajališta aplikacija. Manipulacije se dodatno kompliciraju i postaju neučinkovite, a to na kraju vodi do toga da se ne zna koja vrata pripadaju kojim.



Slika 6. Drugi električni primjer

Ovo je još jedan pokušaj rješenja početnog problema upotrebljavajući BLOB-ove ili *Binary Large OBjects*. BLOB je ustvari string velik jedan byte koji se pohranjuje kao u bazu: aplikacija to razumije ali baza podataka ne. Dijagrami krugova koji se nalaze u stupcu «Layout» predstavljaju BLOB-ove.

Problem kod BLOB-a je taj da se ne mogu vršiti ispitivanja na njima: dali se četveroulazna vrata sastoje od dvoulaznih? BLOB sadrži odgovor ali ne u formatu u kojem bi ga baza podataka mogla prepoznati. Uz to baza ne može napraviti nikakvo ispitivanje tekstualnog sadržaja na BLOB-u.

3.2 «Impedance mismach»

Ranije navedeni primjeri prilično dobro prezentiraju probleme koji se javljaju kod relacijskih baza podataka a u literaturi se navode kao «impedance mismatch». Komponente čije se impedancije ne slažu su aplikacija (u našem slučaju CAD sistem) i baza podataka.

Aplikacije su proceduralne, rade samo sa jednim po jednim podatkom iz skupa podataka i stvaraju kompleksne podatkovne strukture. Baze podataka, na drugu stranu, su deklarativne, rade sa n-torkama iz skupova n-torki i drže podatke u tablicama. Oba pristupa imaju svoju snagu i razloge zašto tako rade, međutim pokušamo li ih udružiti u jednu aplikaciju koja se služi bazom podataka dolazi do ne slaganja.

Rješenje su ponudili zagovaratelji objektno orijentiranih baza podataka. Podatak definiran unutar aplikacije može se pohraniti u bazu podataka bez potrebe ikakve pretvorbe, te se programski jezik aplikacije integrira u bazni DML.

3.3 Svojstva objektne orijentacije

Relacijske baze u pravilu ne dozvoljavaju korisniku da definira vlastiti tip podataka, iako se neki autori ne slažu oko toga, dok objektne baze dozvoljavaju definiciju raznih tipova podataka varijabilne složenosti.

Upotrebljavaju se skupovi, liste, multiskupovi i drugi načini prikazivanja rezultata ispitivanja koje kao rezultat vraća nekoliko objekata.

Metoda je dio koda povezan sa objektom koji ima neograničen pristup stanju tog objekta. Ako želimo primjerice provjeriti dali objekt ima ili nema neko svojstvo, koje se može vidjeti iz njegova stanja, puno je lakše obaviti metodom napisanom za taj slučaj nego praviti upite ovisno o atributima.

Pravilo je generički naziv za dio koda koji se automatski pokreće kada se vrše određene radnje.

Jedna od najvažnijih svojstava objektnih baza je da zajednička svojstva aplikacije i baze osiguravaju dobru komunikaciju između inih.

3.4 Prednosti i mane OODB-a

3.4.1 Prednosti

OODB-e omogućuju korisniku da definira neke svoje apstrakcije dok to kod RDB-a nije moguće. Omogućuju bolju povezanost baza podataka i aplikacija, odnosno do sada neki ne rješivi ili komplicirani problemi između baza i aplikacija su riješeni. OODB-e odstranjuju potrebu za definiranjem vlastitih ključeva jer one imaju OID koji se automatski generira i dodjeljuje objektu. Razvili su se tipovi izjednačavanja različitih tipova podataka, i to 1. dva objekta su ista ako imaju isti OID, 2. dva primitivna objekta su ista ako imaju iste vrijednosti ili 2 ne primitivna objekta su ista ako imaju jednak broj svojstava, 3. kao i drugo pravili samo za svojstva a ne za objekte, 4. isto kao i prvo samo za svojstva a ne za objekte. OODB su razvile objektnu algebru.

3.4.2 Mane

Mana OODB je to što relacijske i objektne baze podataka ne mogu međusobno razmjenjivati podatke. Slijedeća mana OODB i jedan od najvećih problema je optimizacija deklarativnih upita. Standardna algebra upita još je jedna mana OODB a što također za sobom povlači i prije navedenu manu optimizacije. Mnoge OODB pate zbog složenosti upita i najčešće jezik upita nije ANSI SQL kompatibilan. Većina OODB-a ne podržavaju autorizacije pa je ovim putem sigurnost baze dovedena u

pitanje. Dinamička promjena definicija klasa također nije omogućena unutar OODB. Ograničene su mogućnosti poboljšavanja performansi OODB. Ni definicije potpuno funkcionalnih kompleksnih objekata nisu dorađene do kraja. Učinkovitost OODB sveukupno i nije tako jako puno povećana u odnosu na učinkovitost RDB. Postoje i neka svojstva koja OODB-e uopće ne podržavaju. I na kraju, a možda i najvažnije, je to da objektne baze ne leže na čvrstim matematičkim temeljima kao što je to kod relacijskih.

3.5 Što je zapravo OODBMS ?

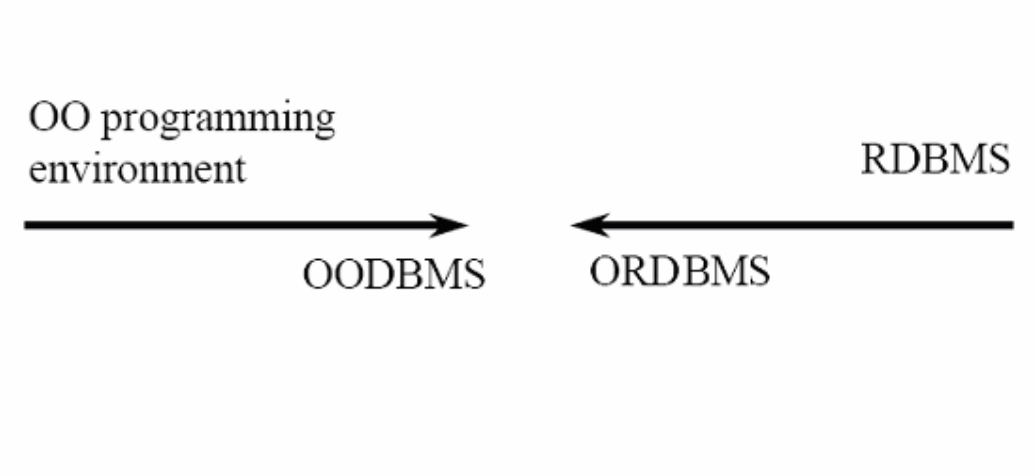
Postavimo li ovo pitanje različitim ljudima dobit ćemo prilično različite odgovore, međutim postoje 3 bitnija manifesta koja su pokušala riješiti ovo pitanje.

Prvi manifest, **1989: The OO DBS Manifesto**, je rekao: uredu, nitko se ne može složiti o tome što je OODBMS pa ćemo mi zato dati set zlatnih pravila. Ako baza podataka zadovoljava ova pravila možete ju nazvati OO.

Drugi manifest, **1990: Third Generation DBS Manifesto,** koji je predstavio svoj novi set pravila je ubiti bio potpuno proturječan prvom. Naglašavao je da su relacijski sustavi uveli dva vrlo važna pravila (ne proceduralni pristup i neovisnost podataka) te bi bilo loše kada bismo ih izbacili iz buduće uporabe. SQL, definiran kao intergalaktički podatkovni jezik, morao se podržati.

Treći manifest, **1995: The Third Manifesto,** puno formalniji i tehnički dotjeraniji, predstavio je sistem čiji su korijeni izlazili iz relacijskih sustava. Rekao je da su u prvom manifestu pogriješili što su ignorirali relacije, dok su u drugom manifestu to dobro shvatili ali su pogriješili u tome što su podupirali SQL koji predstavlja čistu perverziju osnovnih relacijskih ideja. Međutim u trećem manifestu proširuje se relacijski model tako što se dopušta definicija novih tipova podataka za domene.

3.6 Različiti pristupi OODBMS



Slika 7. Različiti pristupi OODBMS

Jedna skupina objektnih baza podataka (lijeva strana dijagrama na slici7) počela se razvijati od početka bez ikakvih veza sa relacijskim sistemima. Svoj tip sistema bazirali su na objektno orijentiranim programskim jezicima te ih razvili tako da su svojstva baza uključili u njih. Ovaj pristup opisan je u prvom manifestu.

Druga grana objektno orijentiranih baza podataka počela se razvijati iz relacijskih sistema te su u njih dodali objektna svojstva, kao što je to opisano u drugom i trećem manifestu. Ovaj pristup opisan je kao objektno relacijski DBMS.

Oba dva pristupa zapravo teže istoj točki, ali dolaze iz različitih smjerova. OODBMS kreće iz objektnog sučelja kojem dodaje mogućnosti za opis deklarativnih upita. ORDBMS kreće iz relacijskih sučelja i proširuje ih na sistem sa objektima.

3.7 Zaključak

Gotovo svatko tko radi na ovom polju priznaje postojanje «impedance mismatch» problema i slaže se da relacijske baze nisu podobne za određene klase aplikacija. Međutim, iako izgleda da objektna tehnologija omogućava rješenje ovog problema, nema zajedničkog zaključka o tome kako bi idealni ODBMS trebao izgledati, čak ni sa teoretskog stajališta (primjer su manifesti).

Glavne prednosti objektnih baza podataka je njihov rich type sistem koji zadovoljava po snazi i zadržava strong typing koji sadrže njegovi krojeni koji se nalaze u OO programskim jezicima, te učinkovitost koju nude u pojedinim aplikacijama gdje su relacijske baze zakazale.

Glavni su nedostaci što one, za razliku od relacijskih, ne leže na čvrstim matematičkim temeljima te je, djelomično i kao posljedica toga, stupanj razvijenosti inih danas prilično mali.

Konstantno se stvaraju novi proizvodi koji se bave bazama podataka, stari obnavljaju, jednako i u industriji i u znanosti, međutim ne postoji neki dogovoreni sustav bodovanja prema kojem bi ih mogli međusobno usporediti.

4 Literatura

[1] Sikha Bagui (2003): "Achievements and Weaknesses of Object-Oriented Databases", Department of Computer Science, University of West Florida, U.S.A.

[2] Frank Stajano (1998): "A Gentle Introduction to Relational and Object Oriented Databases ", Olivetti & Oracle Research Laboratory, Cambridge, U.K.

[3] Adam Stevenson (2000) "Object Oriented Databases"

[4] Ian Holyer (2004) " Object Oriented Databases "