UNIVERZA V LJUBLJANI

EKONOMSKA FAKULTETA

MAGISTRSKO DELO

AKTIVNO PODATKOVNO SKLADIŠČE KOT PLATFORMA ZA OPERATIVNO ODLOČANJE V PODJETJU

Ljubljana, februar 2007 Aleksander Sandy Kucler

IZJAVA

Študent Aleksander Sandy Kucler izjavljam, da sem avtor tega magistrskega dela, ki sem ga napisal pod mentorstvom prof. dr. Jurija Jakliča in skladno s 1. odstavkom 21. člena Zakona o avtorskih in sorodnih pravicah dovolim objavo magistrskega dela na fakultetnih spletnih straneh.

V Ljubljani, dne 23.2.2007 Podpis: _____

KAZALO VSEBINE

KAZALO SLIK

KAZALO TABEL

1 UVOD

1.1 OPREDELITEV PROBLEMA IN NAMEN MAGISTRSKEGA DELA

Poslovna inteligenca predstavlja informacijsko infrastrukturo v organizaciji, s pomočjo katere je možno podatke iz poslovanja pretvoriti v koristne informacije in z njimi sprejemati strateške in taktične odločitve. Uporabniki poslovne inteligence lahko na ta način spremljajo zgodovino dogodkov in ugotavljajo, zakaj se je nekaj zgodilo. Dosegljivost natančnejših, tekočih in primernih informacij jim tako omogoča sprejemanje boljših odločitev. Pravilne odločitve pa se v poslovnem okolju med drugim kažejo kot povečanje dobička, zmanjševanje stroškov, izboljšan odnos s strankami in zmanjšanje poslovnega tveganja (Loshin, 2003, str. 2).

Do nedavnega je še veljalo, da je uporaba poslovne inteligence domena ožje skupine znotraj organizacije (vodstvo, analitiki). Ti poslovno inteligenco uporabljajo za strateške in taktične odločitve. Ažurnost podatkov v tem primeru ni najpomembnejša, saj je za planiranje dolgoročne strategije vseeno, kaj se je v podjetju zgodilo nekaj ur nazaj.

Konkurenčno okolje narekuje organizacijam, da hitro reagirajo na spremembe v poslovnem okolju, zato se čedalje bolj pričakuje, da poslovna inteligenca pomaga ne samo pri strateških, ampak vse bolj tudi pri operativnih odločitvah, ki jih vsakodnevno opravljajo posamezniki v organizaciji. Ta pristop se imenuje operativna poslovna inteligenca.

Cilj operativne poslovne inteligence je pravočasno priskrbeti informacije za operativne poslovne odločitve. To pa je možno edino tako, da se zmanjša čas med poslovnim dogodkom in trenutkom, ko je podatek za ta dogodek na voljo za analize. V idealnem primeru se na ta način poveča dostopnost podatkov do širše skupine uporabnikov, tako da se poslovna inteligenca iz orodja analitikov spremeni v orodje celotne organizacije (Brobst, 2006).

Operativna poslovna inteligenca se v organizacijah kaže kot povečanje učinkovitosti operativnih procesov. Srečamo jo na področjih, kot so detekcija zlorab, elektronsko trgovanje, klicni centri, rezervacijski sistemi ipd. Za vsa omenjena področja je značilno, da je v njih prisotno operativno odločanje na

podlagi podatkov o preteklih dogodkih. Potreba po preteklih podatkih se razlikuje od primera do primera.

Ker je cilj operativne poslovne inteligence zagotovitev informacij za operativno odločanje, je potrebno zmanjšati časovni zamik med analitičnimi in transakcijskimi informacijskimi sistemi v organizaciji. Pri tem se največ problemov pojavlja pri vzpostavitvi ustrezne tehnične infrastrukture, katerega glavni del je sistem za upravljanje baz podatkov. Ta predstavlja platformo, na katerem je postavljeno podatkovno skladišče – osrednja zbirka podatkov za celotno organizacijo. Namenjeno je shranjevanju in dostopanju podatkov za informiranje in odločanje ter je ločeno od transakcijskih informacijskih sistemov. Podatkovno skladišče operativne poslovne inteligence se od običajnega podatkovnega skladišča loči predvsem v tem, da predstavlja kritičen del informacijske infrastrukture v organizaciji, saj je množica operativnih odločitev odvisna prav od podatkov, ki so shranjeni v podatkovnem skladišču. Tako podatkovno skladišče se imenuje aktivno podatkovno skladišče. Od njega se pričakuje visoka zmogljivost, nadgradljivost, razpoložljivost ... uporaba pa mora biti povezana s čim manjšimi operativnimi stroški (Hager, 2006, str. 28).

Namen magistrskega dela je analizirati in predlagati rešitve za probleme, ki se pojavljajo pri postavitvi tehnične infrastrukture. Ugotoviti želim, kakšne naj bodo značilnosti sistemov, ki podpirajo operativno poslovno inteligenco. Pri tem bom izhajal iz razlik med običajno strateško in novo operativno poslovno inteligenco. Pri reševanju problemov želim večino pozornosti posvetil najpomembnejšemu delu operativne poslovne inteligence – podatkovnemu skladišču. Podatkovno skladišče tako postaja nenadomestljiv del celotne organizacije, saj aktivno sodeluje pri izvajanju poslovnih procesov.

1.2 CILJI MAGISTRSKEGA DELA

Cilji magistrskega dela so na podlagi teoretičnih spoznanj in praktičnih izkušenj:

- preveriti stanje na področju operativne poslovne inteligence,
- analizirati poslovne in tehnološke zahteve, ki jih srečamo pri implementaciji aktivnega podatkovnega skladišča, in predlagati rešitve ugotovljenih problemov,
- na primeru preveriti ugotovljene rešitve. Primer bo prototip podatkovnega skladišča, postavljenega na sistemu za upravljanje baz podatkov Oracle 10g, kjer bom izvedel podrobno analizo funkcionalnosti. Na ta način želim preveriti hipotezo, da je Oracle 10g primeren za uporabo na področju aktivnega podatkovnega skladiščenja.

1.3 METODE PREUČEVANJA IN ZASNOVA DELA

Osnovno vodilo pri izdelavi magistrskega dela je od splošnega h konkretnemu. V delu želim postopoma, vendar celovito, predstaviti opisano problematiko. Uporabil bom naslednje metode dela:

- analiza in identifikacija zahtev operativne poslovne inteligence ter
- sinteza zbranih spoznanj in njihova uporaba na primeru analiziranja sistema za upravljanje baz podatkov Oracle 10g za potrebe aktivnega podatkovnega skladiščenja.

Pri izdelavi dela si bom pomagal s strokovno literaturo domačih in tujih avtorjev, temu pa bom dodal tudi lastno znanje in praktične izkušnje, ki sem si jih pridobil med magistrskim študijem in s praktičnim delom na področju poslovne inteligence.

Magistrsko delo sem razdelil na štiri pomembnejše sklope. Prvo poglavje predstavlja problematiko, namen in cilj magistrskega dela. Opisane so metode dela.

V drugem poglavju sem se posvetil poslovni inteligenci. Izhajal sem iz opredelitev, ki so na voljo v literaturi. Osnova poslovne inteligence je proces prehajanja podatkov v znanje. Najpomembnejši del tega prehajanja predstavljajo informacije. V preteklosti se je razvilo več načinov upravljanja z informacijami, ki sem jih kronološko opisal. Poglavje se nadaljuje z opisom običajnih gradnikov poslovne inteligence, med katerimi je obvezno podatkovno skladišče, ki se že na prvi pogled razlikuje od transakcijskih sistemov. Zadnji del poglavja predstavlja najnovejši pristop na področju poslovne inteligence – operativno poslovno inteligenco, njene značilnosti in cilje, ki jih poizkuša reševati v poslovnem svetu.

Podatkovno skladiščenje se je v preteklosti razvijalo zaradi vedno večjih poslovnih potreb. V tretjem poglavju sem ta razvoj opisal skozi pet evolucijskih faz. Vsako podatkovno skladišče se vedno nahaja v eni izmed teh faz. Peta faza predstavlja aktivno podatkovno skladišče, katerega razlike v primerjavi z običajnim podatkovnim skladiščem sem razložil s primerjavo topologij. Osnovni proces nad podatkovnim skladiščenjem je proces polnjenja podatkov. Obstajajo štirje glavni načini polnjenja, ki sem jih primerjalno opisal, vsakega s svojimi prednostmi in slabostmi. Poglavje se nadaljuje z opisom nekaj praktičnih primerov iz poslovnega sveta, katerih skupna lastnost je, da so z uporabo aktivnega podatkovnega skladišča na različne načine izboljšali poslovanje podjetja. Poglavje se zaključuje s tehnološkimi zahtevami, ki morajo biti izpolnjene za potrebe implementacije aktivnega podatkovnega skladišča. Tehnološke zahteve sem razdelil na tri glavne skupine in za vsako izmed njih opisal načine, kako to zagotoviti.

Četrto, tj. zadnje, poglavje je posvečeno analizi primera, ki na praktičnem primeru prototipa poizkuša razložiti potrebe aktivnega podatkovnega skladišča. Prototip internetne prodaje je bil postavljen nad sistemom za upravljanje baz podatkov Oracle 10g. Poglavje se začne s kronološkim pregledom razvoja funkcionalnosti baze Oracle, nadaljuje se z opisom prototipa, na katerem sem poizkusil razložiti specifike, prednosti in pomanjkljivosti sistema za upravljanje baz podatkov Oracle 10g. Poglavje se konča s sklepno analizo.

2 OPERATIVNA POSLOVNA INTELIGENCA

2.1 OPREDELITEV POSLOVNE INTELIGENCE

Izraz »poslovna inteligenca« vsebuje dva različna pomena besede inteligenca.

Prvi, manj pogost, pomen se nanaša na človeško inteligenco, ki je vsebovana v poslovnem okolju. Človeška inteligenca pomeni razum, ki med drugim vsebuje sposobnosti razmišljanja, načrtovanja, reševanja problemov, abstraktnega razmišljanja, razumevanja kompleksnih pojmov, hitrega učenja in učenja iz preteklih izkušenj (Arvey, 2004).

Drugi pomen besede inteligenca pa se nanaša na vrednotenje informacij glede na njihovo pomembnost in veljavo. Če to vrednotenje prestavimo v poslovno okolje, dobimo izraz poslovna inteligenca. Obstaja več definicij poslovne inteligence:

- **je rezultat**, ki je posledica podrobne analize podatkov, pridobljenih iz poslovnih dogodkov. Okolje vsebuje bazo podatkov, poslovne programe in analitične izkušnje. Poslovna inteligenca se včasih enači s sistemi za podporo odločanju, vendar pa v resnici predstavlja precej več, saj med drugim obsega tudi področja, kot so upravljanje z znanjem, planiranje podjetniških virov, podatkovno rudarjenje (California State University Monterey Bay, 2006);
- **je znanje**, ki se je pridobilo iz analize informacij v organizaciji (University of California Santa Cruz, 2006);
- **je obsežen nabor tehnologij** za zbiranje, obdelavo in analiziranje podatkov z namenom, da se poslovnim uporabnikom zagotovijo informacije za sprejemanje boljših poslovnih odločitev. Dobro poslovno odločanje potrebujejo poglobljeno znanje vseh dejavnikov, ki lahko vplivajo na poslovanje organizacije. Ti dejavniki so kupci, tekmeci, poslovni partnerji, zunanje ekonomsko okolje in notranji poslovni procesi (Wikipedia, 2006).

Glede na razlike v definicijah se opazi, da gre pri poslovni inteligenci za večdimenzionalni koncept, ki se lahko tudi kaže kot (Vitt, 2002, str. 13):

- hitrejše sprejemanje boljših odločitev,
- spreminjanje podatkov v znanje,
- podpora racionalnim odločitvam.

Hitrejše sprejemanje boljših odločitev

Organizacije sprejemajo dvoje vrst odločitev: redke velike strateške odločitve, ki so običajno domena vodstva, in nešteto majhnih dnevnih odločitev, ki jih dnevno sprejemajo ostali zaposleni v organizaciji. Zaradi pomembnosti je običajno, da organizacije posvečajo veliko pozornost strateškim odločitvam. Pri njihovem sprejemanju se običajno zbere mnogo informacij, ki se jih obdela s podrobno analizo. Vsaka od variant je natančno ovrednotena. Strateška narava teh odločitev in dejstvo, da so sprejete s strani vodstva, so za organizacije zadosten razlog, da investirajo sredstva z namenom čim boljšega pridobivanja informacij.

Učinkovitost organizacije pa je po drugi strani močno odvisna od množice majhnih odločitev, ki jih dnevno sprejemajo ostali zaposleni na vseh nivoji organizacije. Te majhne operativne odločitve vsaka zase ne spreminjajo strategije podjetja, seštete skupaj pa močno vplivajo na uspešnost poslovanja organizacije in uresničevanje strategije. Zaposleni potrebujejo leta, da si pridobijo izkušnje, znanje in občutek za odločanje. Nekateri izmed njih teh lastnosti nikoli ne pridobijo.

Obstaja več rešitev, ki omogočajo hitrejše sprejemanje boljših odločitev (Liautaud, 2001, 96 str.):

- Vodstvo odloča o vseh odločitvah. To vodi v stresne situacije, zaostanke in opotunitetne stroške, ki nastajajo zaradi neopravljanja strateškega dela.
- Najemanje zunanjih strokovnjakov, kar lahko vodi v nenadzorovano povečanje stroškov.
- Definirajo se pravila, ki pokrivajo vse možne odločitvene primere. Organizacija tako postane toga, kar ni primerno za hitro spreminjajoče se poslovno okolje.
- Vodstvo zaposlenim omogoči način, s katerim je možno boljše odločanje. To je optimalna varianta, saj se organizacija prebudi in postane bolj okretna.

Spreminjanje podatkov v znanje

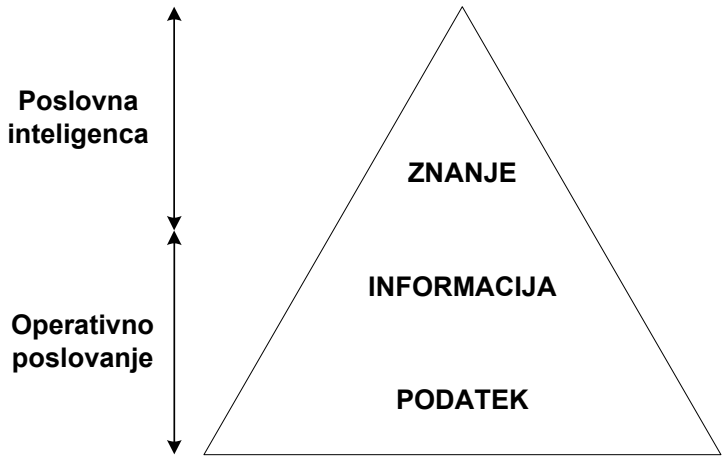
Proces spreminjanja podatkov v znanje poteka preko informacij. Oxfordov slovar (Oxford Reference, 2006) definira naslednje pojme:

- Podatek je statično dejstvo. Namenjen je sklicevanju, upravljanju ali računanju.
- Informacija postane takrat, kadar nam nek podatek lahko služi pri sprejemu odločitve.
- Znanje predstavljajo dejstva, informacije in sposobnosti, ki si jih oseba pridobi z učenjem.

Glede na definicije se vidi, da se znanje nahaja v ljudeh. Ni ga mogoče upravljati tako kot podatke. Upravlja se lahko samo okolje, ki podpira izmenjavo informacij za ustvarjanje znanja.

Medsebojno soodvisnost podatkov, informacij in znanja je možno predstaviti tudi v obliki piramide na sliki 1. Piramido sestavljajo trije elementi: podatki, informacije in znanje. Proces preoblikovanja množice podatkov v informacije je v piramidi viden kot področje, ki pokriva operativno poslovanje. To se v praksi kaže kot nadzor obstoječih delovnih procesov, katerih rezultat so izdelki in storitve. Proces preoblikovanja informacij v znanje – kar predstavlja poslovno inteligenco – pa spreminja poslovne procese in proizvaja nove procese za dosego konkurenčnosti (Hackathorn, 2001).

Slika 1: Piramida podatek/informacija/znanje



Vir: Hackathorn, 2001

Prehajanje podatkov v znanje ni samoumeven proces. Ker se znanje nahaja v ljudeh, je nujno, da so pri tem udeleženi vsi zaposleni v podjetju in ne le vrhovni management. Zaposleni v organizaciji se morajo zavedati, da uporaba njihovega znanja lahko vpliva na uspešnost organizacije, kar se v praksi izraža kot povečanje prodaje, povečanje dobička, zmanjšanje stroškov ipd. Cilj je povečati

inteligentnost poslovanja. Z inteligentnim poslovanjem se lahko sprejemajo boljše in hitrejše odločitve, kar vpliva na povečanje prednosti pred ostalo konkurenco.

Podpora racionalnim odločitvam

V današnjem poslovnem svetu se vedno več odločitev sprejema na podlagi podrobnih racionalnih (razumskih) analiz. O racionalnih odločitvah govorimo, kadar ljudje uporabljajo svoje znanje (ne intuicijo) na posameznem področju kot osnovo za sprejemanje odločitev. Tipično racionalno odločanje je izbiranje najboljše izmed več alternativ. Vsaka izmed alternativ je sestavljena iz več kriterijev. Zato običajen proces razumskega odločanja poteka po naslednjih korakih (Bazerman, 2001):

1. definiranje problema
2. identifikacija kriterijev
3. določanje teže kriterijev
4. definiranje alternativ
5. ovrednotenje posameznih alternativ na podlagi kriterijev
6. izbira najustreznejše alternative

Zahtevnost postopka zahteva od ljudi, da so zbrani pri odločevanju. Podjetja se nahajajo v tekmovalnem okolju, zato so odločevalci v podjetjih podvrženi različnim pritiskom, kot so pomanjkanje časa, strah in stres. Poleg tega na odločanje vplivajo še:

- omejena racionalnost odločevalcev,
- subjektivnost in težka merljivost kriterijev,
- tveganje in negotovost,
- vpletanje čustev.

Določene raziskave (Klein, 1993) kažejo, da odločevalec pod pritiskom ne odloča racionalno, temveč naravno. Pri naravnem odločanju odločevalec izbere alternativo, ki je prva prepoznana za najbolj primerno. Zaradi tega je rezultat naravnega odločanja močno odvisen od izkušenj odločevalca.

Poslovna inteligenca lahko pri odločanju sodeluje tako, da odločevalcu pomaga pri izbiri najustreznejše alternative. Tako so odločitve tudi v trenutkih različnih pritiskov izbrane racionalno.

2.2 UPRAVLJANJE Z INFORMACIJAMI

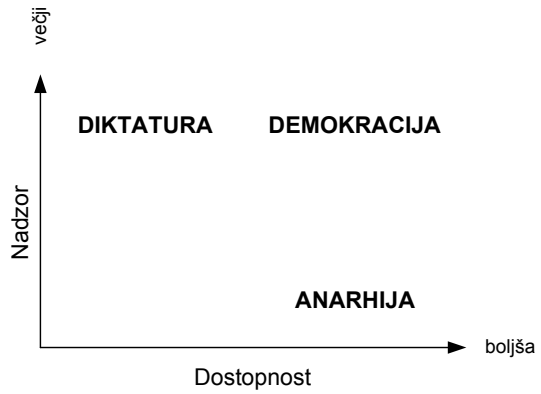
Prejšnje poglavje je opredelilo osnovne pojme znotraj poslovne inteligence. Najpomembnejši izmed njih je vsekakor informacija. Informacija predstavlja

osnovo za odločanje. Kvaliteta odločanja pa je odvisna od načina upravljanja z informacijami. Zgodovino upravljanja z informacijami znotraj organizacij lahko predstavimo s štirimi modeli, ki si kronološko sledijo v naslednjem vrstnem redu (Liautaud, 2001):

- informacijska diktatura,
- informacijska anarhija,
- informacijska demokracija in
- informacijska ambasada.

Modeli se medsebojno razlikujejo v dveh glavnih kriterijih: nadzoru in dostopnosti do informacij. Slika 2 v obliki grafa prikazuje razporeditev omenjenih modelov upravljanja z informacijami glede na dostopnost in nadzor nad informacijami. Ker graf prikazuje načine upravljanja v organizaciji, je iz njega namenoma izpuščen model informacijske ambasade. Ta predstavlja model upravljanja z informacijami izven meja organizacije.

Slika 2: Modeli upravljanja z informacijami



Vir: Prirejeno po Liautaud, 2001

Informacijska diktatura

Model informacijske diktature izhaja iz 80. let prejšnjega stoletja. Za to obdobje je bilo značilno, da je bila moč informacij dostopna majhnemu številu ljudi, ki so predstavljali vrhovni management podjetja. Čeprav je to zastarel model upravljanja z informacijami, ga v določenih podjetjih opažamo še danes.

Tehnološko gledano je za model informacijske diktature značilna uporaba velikih računalnikov, na katerih so nameščeni namenski programi za podporo poslovnim procesom. Takšno okolje je tipična oblika centraliziranega okolja, kjer se podatki zajemajo, shranjujejo in obdelujejo na enem mestu – centralnem računalniku. Za dostop do informacij se uporabljajo posebni sistemi, ki se imenujejo direktorski informacijski sistemi. Direktorski informacijski sistemi so kompleksni računalniški

programi, ki temeljijo na paketnih obdelavah in preddefiniranih poročilih. Njihova slabost se kaže v visoki ceni in v težavnem vzdrževanju.

Informacijska anarhija

Ker je model informacijske diktature močno omejeval dostopnost do informacij, se je v 90. letih prejšnjega stoletja izoblikoval model informacijske anarhije. Vzporedno z razvojem novih tehnologij, kot so osebni računalniki, namizne baze podatkov, urejevalniki teksta in preglednice, se je povečevala skupina uporabnikov, ki so imeli dostop do informacij. V to jih je silila tudi čedalje močnejša konkurenca. Podatki so se iz centralnega računalnika, ki je bil značilen za model informacijske diktature, razširili po celotni organizaciji. Oblikovati so se pričeli podatkovni silosi – hrambe podatkov po posameznih področjih organizacije (prodaja, marketing, finance, proizvodnja ...). Pojavili so se problemi neintegriranosti, pomanjkljivosti, podvojenosti in nenatančnosti podatkov. Čeprav so se uporabniki tega modela zavedali teh problemov, so bile rešitve redke, saj so bili podatki razpršeni na različnih in med seboj pogosto nezdružljivih platformah.

Informacijska demokracija

V zadnjih letih čedalje več podjetij spoznava koristi zagotavljanja informacij svojim zaposlenim. Izkazalo se je, da se prožnost podjetja poveča predvsem s tem, da imajo zaposleni dostop do informacij. Model informacijske demokracije predvideva, da se morajo odločitve sprejemati tam, kjer se tudi izvajajo – na najnižjih nivojih organizacije. Kvalitetne odločitve pa se lahko sprejemajo samo ob prisotnosti informacij. Rast uporabe interneta je vplivala na ponudnike programskih rešitev tako, da so ti prilagodili svoje aplikacije za uporabo v internetu. Na ta način se je povečala njihova uporaba, saj se aplikacije ne izvajajo več na lokalnih računalnikih, temveč centralno, vendar so aplikacije kljub centraliziranosti zaradi interneta dostopne celotni organizaciji. Podjetje tako deluje bolj inteligentno.

Raziskava podjetja Business Objects (Liautaud, 2001) kaže, da je nivo poslovne inteligence v organizaciji odvisen od treh kriterijev:

- stopnje uporabe poslovne inteligence znotraj organizacije (delež uporabnikov poslovne inteligence glede na število vseh uporabnikov računalnikov),
- stopnje pooblaščanja zaposlenih (števila uporabnikov, ki imajo pravico izvajanja lastnih poizvedb glede na število vseh uporabnikov poslovne inteligence) in
- stopnje pripravljenosti za delitev informacij med različnimi področji v organizaciji.

Rezultat raziskave je pokazal dve pomembni dejstvi:

- Večja ko je stopnja demokratizacije in pooblaščanja zaposlenih, bolj inteligentna je organizacija.
- Večja ko je pripravljenost za podiranje informacijskih pregrad med področji, bolj inteligentna je organizacija.

Pojem inteligentne organizacije je relativno nov. Izhaja s področja upravljavske kibernetike, ki preučuje sposobnost sistemov za preživetje. Inteligentna organizacija temelji na znanju in neprestanem učenju, kar se v praksi kaže kot povečana agilnost in prožnost poslovanja (Yoles, 2006).

Za informacijsko demokracijo ni nujno, da se konča znotraj meja organizacije. Uporaba interneta omogoča, da se določeni podatki širijo tudi izven meja organizacije. Ta model upravljanja z informacijami se imenuje informacijska ambasada.

Informacijska ambasada

Z uporabo interneta je dostop možno razširiti izven meja organizacije tako, da subjekti iz okolja dostopajo do informacij, ki so pomembni za njih. Upravljanje informacij z modelom informacijske ambasade povečuje inteligentnost poslovanja dobaviteljev, kupcev in poslovnih partnerjev. Informacije se do zunanjih uporabnikov širijo preko internetnih portalov in namenskih podatkovnih povezav, kot je npr. medpodjetniško poslovanje (angl. *Business to Business, B2B*). Širjenje informacij izven meja organizacije prinaša:

- prihranek denarja za kupce,
- zmanjšanje števila klicev v klicne centre organizacije,
- povečanje zadovoljstva kupcev in s tem povezane lojalnosti,
- zmanjšanje stroškov zaradi transformacije papirnatih poročil v elektronsko obliko,
- nov vir zaslužka.

S povečanjem inteligentnosti poslovanja zunanjih uporabnikov informacij se posredno poveča inteligentnost same organizacije, saj se medsebojno poslovanje dvigne na višji nivo.

2.3 GRADNIKI POSLOVNE INTELIGENCE

V prejšnjem poglavju omenjena definicija govori, da gre pri poslovni inteligenci za pridobivanje znanja iz zbranih, obdelanih in analiziranih podatkov. Pri procesu prehajanja podatkov v znanje si lahko pomagamo z informacijsko tehnologijo. Ta

nam lahko pomaga z različnimi izdelki, ki jih glede na svojo funkcijo delimo v dve osnovni skupini:

- podatkovno skladišče in
- različna analitična orodja.

2.3.1 Podatkovno skladišče

Podatkovno skladišče je osrednja zbirka podatkov za celotno podjetje. Namenjeno je shranjevanju in dostopanju do podatkov, za informiranje in odločanje ter je kot tako ločeno od transakcijskih sistemov. Zaradi svoje pomembnosti predstavlja najpomembnejšo komponento poslovne inteligence.

Organizacija potrebuje za uresničevanje svojih ciljev nemoten potek poslovnih procesov. Poslovni proces je sestavljen iz dogodkov različnih vsebin, dogodki sami pa si lahko sledijo v različnem časovnem zaporedju. Informacijski sistemi, ki podpirajo delovanje poslovnih procesov, se imenujejo transakcijski sistemi.

Transakcijski sistem predstavlja hrbtenico oziroma kritični del organizacijskih informacijskih sistemov. Njegova naloga je zbiranje, shranjevanje, nadzorovanje in širjenje informacij za vse ponavljajoče se poslovne transakcije. Cilj transakcijskih sistemov je zagotavljanje vseh informacij, ki jih določata notranja organizacijska politika in zunanja zakonodaja.

Podatkovna skladišča se običajno polnijo s podatki iz več virov, ki jih lahko razdelimo na dve skupini:

- notranji viri (transakcijski sistemi) in
- zunanji viri (statistične baze, ostale prostodostopne in plačljive baze podatkov).

Bill Inmon podatkovno skladišče definira kot zbirko podatkov, ki pomaga pri odločanju v organizaciji. Značilnosti so (Inmon, 2005):

- organiziranost po poslovnih področjih,
- integriranost,
- nespremenljivost,
- podatkovno skladišče vsebuje zgodovinske podatke.

Organiziranost podatkov po poslovnih področjih

Transakcijski sistemi običajno vsebujejo podatke, ločene po procesih, npr. kataloška prodaja, prodaja na drobno, veleprodaja. Vsak od teh sistemov lahko

priskrbi podatke o procesu, ki ga podpira. Uporabniki pa običajno potrebujejo podatke, gledane s strani poslovnega področja v celoti, ne glede na to, iz kakšnih virov (procesov) prihajajo. Pod organiziranostjo po poslovnih področjih si prestavljamo npr. prodajo, finance, servis. Organiziranost podatkov glede na poslovno področje, in ne glede na proces, omogoča boljše razumevanje vsebine in s tem boljši pogled na celotno poslovno področje.

Integriranost

Integriranost podatkovnega skladišča pomeni, da so podatki iz različnih virov obdelani tako, da ustrezajo določenim pravilom. To se v praksi kaže kot poenotenje načina šifriranja, dopolnitev manjkajočih podatkov, čiščenje podatkov. Na ta način se odpravlja neskladnost med različnimi vrstami zapisa podatkov in omogoči poenotenje vsebine znotraj celotne organizacije.

Nespremenljivost

Nespremenljivost se v podatkovnem skladišču kaže kot statičnost podatkov. Ko je podatek enkrat shranjen v podatkovnem skladišču, se ta praviloma ne spreminja več. Ta pristop se razlikuje od transakcijskih sistemov, kjer se podatki stalno spreminjajo in tako odsevajo trenutno stanje organizacije.

Vsebovanje zgodovinskih podatkov

Podatkovno skladišče je brez vsebovanja zgodovinskih podatkov neuporabno. Mnogo analiz namreč primerja trenutne in zgodovinske podatke, s čimer se napovedujejo trendi. Zgodovinski podatki se v podatkovnem skladišču shranjujejo odvisno od poslovnih potreb ter v zadnjem času čedalje bolj vpletene zakonodaje – predvsem s področja varstva osebnih podatkov in preprečevanja terorizma. Tako lahko čas shranjevanja podatkov glede na omenjena kriterija v praksi nanese tudi več let ali desetletij.

2.3.2 Analitična orodja

V preteklosti je bilo osnovno orodje za pripravljanje analiz preglednica (npr. Excel). Uporabnikom preglednic je bilo omogočeno vnašanje poljubnih podatkov, izvajanje različnih operacij nad njimi, oblikovanje končnega zgleda poročil in podobno. Razvoj analitičnih orodij za potrebe poslovne inteligence je moral, če je hotel, da bi uporabniki sprejeli novo tehnologijo, upoštevati dotedanji primat preglednic na področju priprave analiz. Zaradi podobnosti danes končni uporabnik mnogokrat ne ugotovi, če je bilo določeno poročilo izdelano s preglednico ali z naprednim analitičnim orodjem. Obstaja več delitev analitičnih orodij. Glede na naloge, ki jih opravljajo, jih lahko delimo na (Wikipedia, 2006):

- orodja za poizvedovanje in izdelavo poročil,

- orodja za sprotno analitično obdelavo podatkov (angl. *OnLine Analytical Processing, OLAP*),
- orodja za podatkovno rudarjenje (angl. *Data Mining, DM*),
- napredna analitična orodja.

2.3.2.1 Orodja za poizvedovanje in izdelavo poročil

Predstavljajo skupino osnovnih orodij, ki se jih običajno najprej implementira pri postavitvi poslovne inteligence. Orodja se medsebojno sicer razlikujejo, po mojih izkušnjah pa je za večino izmed njih značilno, da izpolnjujejo naslednje zahteve:

- dostop do različnih virov podatkov, kamor so všteti vsi vodilni ponudniki relacijskih in večdimenzionalnih baz podatkov ter različni tipi datotek,
- so enostavna za učenje in preprosta za uporabo (uporabljajo grafični vmesnik),
- imajo sposobnost komuniciranja s stavki SQL in uporabo parametrov,
- vsebujejo funkcije za delo s podatki,
- so sposobna obdelovati in sprejemati velike količine podatkov,
- delujejo v odjemalec/strežnik ali večnivojski arhitekturi,
- omogočajo omejevaje/dovoljevanje dostopa do posameznih podatkov glede na uporabniške pravice,
- omogočajo prikaz podatkov v različnih oblikah, kot so tabele, grafi, nadzorne plošče (angl. *dashboard*),
- omogočajo avtomatizacijo izvajanja in izvoza podatkov v razširjene formate, kot so Excel, XML, PDF, TXT ...

2.3.2.2 Orodja za sprotno analitično obdelavo podatkov

O orodju OLAP govorimo, če ta izpolnjuje test FASMI (angl. *Fast Analysis of Shared Multidimensional Information, FASMI*), ki ga sestavlja pet zahtev (Pendse, 2005):

- Hitrost (angl *Fast*). Sistem mora biti dimenzioniran tako, da večino poizvedb vrne v času 5 sekund. Samo izjemoma se lahko poizvedbe izvajajo 20 in več sekund.
- Analitičnost (angl. *Analysis*). Sistem mora uporabniku na preprost način omogočati, da nad podatki izvaja različne logične in analitične operacije. Uporabniku orodja mora biti omogočeno dodajanje novih izvedenih funkcij.
- Omogočanje sočasnega dostopa (angl. *Shared*). Sistem mora sposoben, da v primeru sprememb v podatkih spremeni tudi prikaz teh. Obenem mora

biti omogočena varnost podatkov, kar se določa z dodeljevanjem uporabniških pravic.

- Večdimenzionalnost (angl. *Multidimensional*). Je osnovna zahteva orodij OLAP. Sistem mora omogočati hierarhičen dostop do podatkov in uporabo več dimenzij.
- Sistem mora zagotavljati informacije (angl. *Information*). Količina informacij je odvisna od količine podatkov, kar pa je običajno omejeno z razpoložljivimi računalniškimi kapacitetami.

2.3.2.3 Orodja za podatkovno rudarjenje

Podatkovno rudarjenje predstavlja proces implementacije pristopov umetne inteligence in statističnih metod nad velikimi količinami podatkov z namenom, da se odkrijejo še neodkrite zakonitosti v podatkih. Podatkovno rudarjenje uporablja pristope strojnega učenja, statističnih in vizualizacijskih tehnik z namenom odkrivanja novih korelacij, vzorcev v podatkih in trendov na način, ki je človeku razumljiv (Piatetsky-Saphiro, 1991).

Orodja za podatkovno rudarjenje se med seboj razlikujejo. Njihovo kvaliteto določata predvsem število razpoložljivih algoritmov in možnost neposrednega dostopa do različnih virov podatkov. Predpogoj je, da so podatki neagregirani, torej na nivoju posameznega dogodka. Razlog za to je v tem, da agregacija zamegli vzorce v podatkih, ki bi podatkovnemu rudarjenju služili za odkrivanje novih, še neodkritih, zakonitosti (Biere, 2003).

Pri podatkovnem rudarjenju lahko govorimo o dveh različnih pristopih (Berry et al., 2000):

- neusmerjenem podatkovnem rudarjenju in
- usmerjenem podatkovnem rudarjenju.

Neusmerjeno podatkovno rudarjenje

Neusmerjeno podatkovno rudarjenje išče pravila v obstoječih podatkih. Uporabljamo ga, kadar naš cilj ni določen. V tem primeru podatkovno rudarjenje odkrije vzorce in nam prepusti odločitev o njihovi (ne)pomembnosti. Glavna predstavnika algoritmov sta:

- Gručenje (angl. *clustering*), katerega lastnost je delitev zapisov v različne skupine. Cilj gručenj je poiskati skupine, ki so si medsebojno različne, pri pogoju, da so si predstavniki posamezne skupine (zapisi) medsebojno podobni.

- Povezovalna pravila (angl. *link analysis*), ki iščejo povezave med posameznimi atributi določenega zapisa. Primer uporabe povezovalnih pravil je analiza nakupovalne košarice.

Usmerjeno podatkovno rudarjenje

O usmerjenem podatkovnem rudarjenju govorimo, kadar želimo pri podanih atributih napovedovati vrednost ciljnega atributa. Pristop se uporablja za izdelavo napovedovalnih modelov. Glede na tip ciljnega atributa razlikujemo naslednja dva algoritma:

- klasifikacijo: ciljni atribut je nominalni, in
- regresijo: ciljni atribut je numerični.

Natančnost modela podatkovnega rudarjenja je odvisna predvsem od jasnega definiranja problema, ki ga želimo rešiti. Razpolagati moramo z bogato zbirko podatkov, ki se nanaša na obravnavano tematiko, razumeti jedro podatkov in razpolagati z ustreznimi merami, s katerimi lahko ocenjujemo točnost podatkovnega modela. Ne glede na izbrani pristop, se pri vpeljavi rešitev podatkovnega rudarjenja srečujemo s šestimi koraki (Paul, et al., 2002):

- definiranje problema,
- priprava podatkov,
- gradnja podatkovnih modelov,
- preizkušanje podatkovnih modelov,
- vpeljava modela in
- upravljanje z meta podatki, povezanimi s transformacijo, čiščenjem podatkov, gradnjo modelov in preizkušanjem modelov.

Uporaba v poslovnem svetu

Podatkovno rudarjenje se uporablja na različnih poslovnih področjih. Možno ga je implementirati v vseh fazah upravljanja s strankami, od faze pridobivanja novih kupcev, povečevanja prihodkov od že obstoječih kupcev in skrbi za ohranitev najboljših kupcev. Različne panoge imajo svoje referenčne primere podatkovnega rudarjenja. V telekomunikacijah, poslovanju s kreditnimi karticami, zavarovalnicami in borznem trgovanju se podatkovno rudarjenje uporablja na področju odkrivanja zlorab (angl. *fraud detection*). Medicina ga uporablja pri napovedovanju učinkovitosti posameznih operativnih posegov in postopkov zdravljenja. Finančne institucije ga uporabljajo za napovedovanje prihodnjih tečajnih razmerjih na denarnih in borznih trgih, v trgovini pa ga srečamo pri pospeševanju in napovedovanju prodaje izdelkov in storitev (Berson, 1999; Two Crows, 2006).

2.4 ZNAČILNOSTI OPERATIVNE POSLOVNE INTELIGENCE

2.4.1 Opredelitev

Tradicionalno mišljenje, da poslovna inteligenca obdeluje informacije, ki so se zgodile v preteklosti, danes postaja zastarelo. Ne dolgo nazaj običajno opravilo, kot je analiza dogodkov preteklega meseca, se je spremenilo v tej meri, da se danes pričakuje analiziranje dogodkov pretekle ure ali manj. Podjetja težijo k temu, da bi prednosti poslovne inteligence vključili v operativni proces. Rezultat tega je razširitev kroga uporabnikov poslovne inteligence iz maloštevilnega kroga vodstva in analitikov v ozadju do širokega kroga zaposlenih v prvih vrstah, ki imajo vsakodnevni stik z operativnim odločanjem. Podjetje želi na ta način povečati agilnost in prožnost (Liautaud, 2001).

Kaj je glavni namen poslovne inteligence? Poslovna inteligenca skrbi za dostavo informacij, ki se jih uporablja pri procesu odločevanja.

Strateško, taktično in operativno odločanje

Odločanje je proces, s katerim se vsak izmed nas srečuje v vsakdanjem življenju. Dnevno se odločamo, kaj bomo počeli, kaj bomo jedli, kako bomo organizirali svoj čas. Odločanje se izvaja tako, da se izbira med različnimi alternativami. Tudi odločanje v organizacijah poteka na podoben način. Izvaja se ga tako, da se s pomočjo pridobljenih informacij iz množice alternativ izbere tiste, ki so najprimernejše za dosego organizacijskih ciljev. Odločanje je zato bistvena komponenta, ki vpliva na uspešnost organizacije. Osnovna funkcija vodstva je sprejemanje pravilnih odločitev. Nivoje odločanja v organizacijah lahko delimo na (Nickels, 1999):

- **Strateško odločanje**: izvajajo ga najvišje vodstvene strukture v organizaciji. Strateške odločitve vplivajo na dolgoročno poslovanje organizacije kot celote in kot take običajno potrebujejo daljši čas (nekaj dni, tednov, mesecev), preden iz ideje stopijo v dokončno veljavo. Primeri strateških odločitev so:
    - izbira trgov,
    - izbor izdelkov in storitev, ki jih bo podjetje ponujalo,
    - določanje virov financiranja,
    - glavne kapitalske investicije ipd.

  Skupni lastnosti strateških odločitev sta njihova kompleksnost in pomanjkanje informacij v času sprejemanja odločitev. Odločevalci jih

pogosto sprejemajo na podlagi preteklih izkušenj, deloma tudi z uporabo intuicije.

- **Taktično odločanje**: je srednjeročno usmerjeno. Izvajajo ga srednje vodstvene strukture v organizaciji. Cilj taktičnih odločitev je usmerjanje podjetja k zadanim strateškim odločitvam. Primer taktičnih odločitev je izbira reklamne agencije, ki bo skrbela za promoviranje novega izdelka.
- **Operativno odločanje**: Cilj operativnih odločitev je tekoče izvajanje vsakodnevnih aktivnosti, ki gledane kot celota vodijo k uresničevanju strateških ciljev. Običajno jih izvajajo nižje vodstvene strukture. Operativno odločanje je izmed opisanih treh oblik najbolj definirana oblika odločanja. Zato se ga da pogosto opisati s poslovnimi pravili. Hitrost izvajanja operativnih odločitev se razlikuje med posameznimi poslovnimi primeri, skupno vsem primerom pa je, da so odločitve hitre (od nekaj sekund do nekaj ur). Primeri operativnih odločitev so naročanje materiala, odobravanje kreditov posameznim strankam, predlaganje storitev na podlagi pretekle uporabe le-teh.

Mnogo organizacij ne vsebuje izrazitih hierarhičnih nivojev oziroma v svojem organigramu ne vsebujejo srednjega managementa. V takšnih primerih se običajno taktične in operativne odločitve združijo v en nivo odločanja. Za njihovo izvajanje skrbi nižji management in/ali skupine zaposlenih.

Tabela 1: Primerjava tradicionalne in operativne poslovne inteligence

	Tradicionalna poslovna inteligenca	**Operativna poslovna inteligenca**
Namen	Za strateško in taktično odločevanje (»Kdo so naši najboljši kupci v preteklem mesecu?«)	Za operativno odločanje (»Ali naj tej stranki odobrimo naročilo?«)
Končni uporabniki	Analitiki, vodstvo	Izvrševalci odločitev v hierarhično najnižjih vrstah organizacije, področni vodje, vodstvo, analitiki
Pogostost polnjenja	Dnevno in manj pogosto	Od realnega časa do največ enega dneva
Običajna uporaba	Prodaja, finance, marketing, proizvodnja	Oskrbovalne verige, rezervacijski sistemi, klicni centri

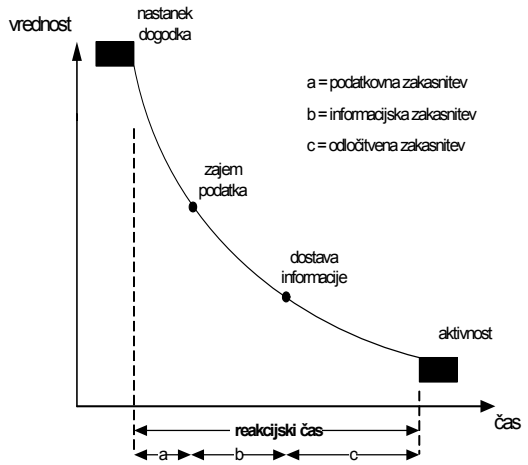
Vir: Hall, 2004

Ne glede na nivo odločanja, ostajajo osnovna pravila enaka. Poslovna inteligenca še vedno predstavlja način odločanja, analiziranja vrednosti, napovedovanja

trendov, združevanja rezultatov in dostave informacij. Operativna poslovna inteligenca opravlja vse te omenjene naloge in več. Njena skrb je zagotavljanje informacij za podporo operativnim odločitvam. Tabela 1 prikazuje glavne točke, v katerih prihaja do razlikovanja tradicionalne in operativne poslovne inteligence.

S tem ko dostavlja informacije vsem nivojem v podjetju, se pokaže osnovna zahteva, to je skrb za zagotavljanje pravočasnosti informacije. Danes se podjetja nahajajo v tekmovalnem okolju, zato hitrost dostopa do informacij pogosto predstavlja glavni vzrok za uspeh ali poraz pri izpeljavi določenega posla. Vrednost informacije ni konstanta, ampak se manjša v odvisnosti od časa, ki preteče od nastanka dogodka do trenutka, ko je informacija na voljo odločevalcem.

Slika 3: Vrednost informacije glede na čas nastanka



Vir: Hackathorn, 2002

Slika 3 prikazuje zvezo med vrednostjo informacije in časom. Čas od nastanka dogodka do izvedbe odločitve se imenuje reakcijski čas. V tem času vrednost informacije eksponentno pada. Reakcijski čas je vsota časov treh zakasnitev:

- **Podatkovna zakasnitev** predstavlja čas med nastankom dogodka in časom, ko je podatek na voljo v podatkovnem skladišču.
- **Informacijska zakasnitev** predstavlja čas od začetka analize, shranjevanja rezultata in dostave do končnega uporabnika.
- **Odločitvena zakasnitev** je čas, potreben za razumevanje informacije in sprejetja ustrezne akcije.

Cilj vsakega odločanja je zmanjšati vse tri čase, ki sestavljajo reakcijski čas. V vsakdanjem življenju se je v letih utrdilo prepričanje, da je odločitvena zakasnitev nedotakljiva pravica odločevalcev in je zato za zmanjševanje reakcijskega časa mogoče zmanjševati samo podatkovno in informacijsko zakasnitev.

2.4.2 Cilji operativne poslovne inteligence

Za organizacije je pomembno, da so informacije dostavljene pravočasno (angl. *right-time*). Pojem pravočasnosti na sliki 3 definirata podatkovna in informacijska zakasnitev. V preteklosti je poslovno inteligenco v povprečju uporabljalo 5 % ljudi v organizaciji, ki so imeli dostop do 10 % podatkov v organizaciji. Čas, ki je definiral pravočasnost, je bil daljši. Cilj operativne poslovne inteligence je zmanjšati čas od nastanka dogodka do dostave informacije do te mere, da je ta informacija dostavljena dovolj hitro tudi za potrebe operativnega izvajanja procesov. Posledica tega je skrb za zmanjševanje podatkovnih in informacijskih zakasnitev. Tudi pri operativni poslovni inteligenci se pravočasnost dostavljenih informacij razlikuje od enega do drugega poslovnega primera. V nadaljevanju bom opisal nekaj izmed njih (TDWI, 2006).

Primer iz letalskega prometa (Continental Airlines)

Letalska družba uporablja operativno poslovno inteligenco. Sistem je postavljen tako, da beleži vse dogodke, ki bi lahko vplivali na poslovanje (vremenski pogoji, zastoji ...). Letalska družba je pravočasno informacijo definirala kot tisto, ki opisuje dogodek z nastankom v zadnjih petih minutah. Zato se podatkovno skladišče polni v intervalih, manjših od pet minut.

Če je na primer v zraku letalo, je v primeru zamude pristanka zaradi vremenskih težav osebje na letališču obveščeno, da je nekaj izmed teh potnikov takšnih z visoko boniteto. Te potnike obravnava drugače kot ostale, razlikovanje je najbolj očitno v primeru, če imajo ti potniki rezervirane povezane polete, ki pa jih bodo zaradi zamude letala zamudili. Še pred pristankom letala se jih obvesti, da imajo prestavljen naslednji let.

Za letalske družbe je značilno to, da za pogoste regionalne polete izdajajo več letalskih kart, kot ima letalo sedežev. To počnejo namenoma, saj statistično gledano obstaja določeno število potnikov, ki ne izkoristijo vnaprej kupljene vozovnice. Običajna poslovna praksa je, da so cene vozovnic dražje, bolj ko se bliža dan poleta. Cene zgodnjih vozovnic so tako lahko nekajkrat cenejše od vozovnic, kupljenih na dan poleta. Letalske družbe težijo k 100% zasedenosti poletov. Na ta način optimizirajo prihodke. V primeru, da je potnikov za določeni polet vseeno preveč, lahko osebje na podlagi profila obnašanja potnika v zgodovini določi, ali je to potnik, ki ga je potrebno spustiti na letalo, oziroma potnik, kateremu se ponudi finančno nadomestilo in bo v letališčem hotelu z veseljem počakal do naslednjega poleta.

Primer iz farmacije (Merck-Medco)

Podjetje, specializirano za on-line izdajanje zdravil, uporablja operativno poslovno inteligenco za spremljanje poslovanja. Vsak naročnik, ki preko internetne strani naroča zdravila, ima zaradi vključene operativne poslovne inteligence možnost vpogleda v svojo zgodovino nakupov zdravil. S pomočjo poslovnih pravil, temelječih na podatkovnem rudarjenju, je sistem razširjen do te mere, da je sposoben na podlagi zgodovinskih naročil predvidevati, če je novo naročilo zdravil združljivo s prejšnjim. Na ta način želi podjetje preprečiti napačno predpisovanje zdravil oziroma izdajo zdravil, ki bi v kombinaciji z ostalimi, ki jih bolnik že uporablja, povzročila navzkrižno reakcijo in s tem ogrozila bolnikovo življenje.

Primer iz igralništva (Harah's Entertainment)

Igralci v 25 casinojih s skupno več kot 50.000 igralnimi avtomati uporabljajo kartico zvestobe, ki jim omogoča različne ugodnosti znotraj casinojev. Ta kartica pa obenem služi kot identifikator, s pomočjo katerega je možno spremljati navade posameznega igralca (vzorec igranja, količina denarja v igri ...). Podatki se z nekajsekundnim zamikom shranjujejo v sistem, kjer so na voljo operativni poslovni inteligenci. Tako je možno igralca, ki ima slab dan v igralnici (kar je relativno od posameznika do posameznika), ob ustreznem trenutku razveseliti s pozornostjo, ki je odvisna od relativne višine zaigranega denarja. Igralec tako za tolažbo dobi od brezplačne pijače, kosila do bona za brezplačno bivanje v hotelih igralniške verige. Ta bon lahko izkoristi pri naslednjem obisku.

Uvedba operativne poslovne inteligence se je že v prvem letu delovanja izkazala za pravilno odločitev. Ugotovilo se je povečanje različnih kazalnikov, med njimi:

- povečanje navzkrižne prodaje znotraj igralniške verige za 72 %,
- povprečno število obiskov posameznega igralca se je povečalo z 1,2 na 1,9 na mesec,
- na nivoju podjetja se je dobiček povečal za 50 milijonov USD.

Primerov iz prakse je še veliko. Za vse pa je značilno, da so informacije vredne le, če so dostavljene pravočasno. Organizacije so pripravljene tolerirati samo odločitveno zakasnitev, ki pa jo prav tako želijo zmanjšati. Na zmanjšanje odločitvene zakasnitve se lahko vpliva na tri načine:

- Z **alarmiranjem** odločevalcev. Sistem mora biti sposoben prepoznati nenavaden poslovni dogodek ali stanje. Običajno delovanje odločevalca je potrebno prekiniti (telefonski klic, SMS, pisk , alarm ...) in ga preusmeriti na ta dogodek.
- Odločevalec mora biti **informiran**. Sistem mora biti sposoben predstaviti analizo dogodka tako, da je odločevalec hitro sposoben razumeti dogodek.

Analiza mora biti izvedena tako, da je odločevalec sposoben postaviti prioritete izvajanja odločitve.

- Odločevalec mora biti **voden**. Sistem mora biti sposoben predlagati primerno akcijo za posamezni dogodek.

3 ZAHTEVE AKTIVNEGA PODATKOVNEGA SKLADIŠČA ZA POTREBE OPERATIVNE POSLOVNE INTELIGENCE

3.1 EVOLUCIJA PODATKOVNEGA SKLADIŠČENJA

3.1.1 Management uspešnosti in učinkovitosti

Aktivno podatkovno skladišče je potrebno obravnavati kot rezultat evolucije poslovnega okolja. To vsebuje več dejavnikov, ki silijo organizacije k zagotavljanju boljših odločitev. Najpomembnejši izmed njih so (Gilmore, 2000):

- konkurenca,
- kupci,
- regulatorji trga in zakonodaja ter
- tehnologija in internet.

Tehnološki napredek, vključno z internetom, spreminja področje delovanja organizacij in s tem podatke, ki so koristni pri sprejemanju odločitev. Regulatorji trga in zakonodaja narekujejo, kakšni podatki se lahko zbirajo, shranjujejo in uporabljajo. Pričakovanja kupcev po njim prilagojenih storitvah in konkurenčno okolje silijo organizacije, da se odločajo na individualni osnovi za vsako stranko posebej. Spreminjanje okolja vpliva na poslovne procese. Zato se ti nenehno spreminjajo. Njihovo nenehno spreminjanje pogosto vodi v neoptimiziranost poslovanja.

Management uspešnosti in učinkovitosti (angl. *Business Performance Management, BPM*) sestavlja množica procesov, katerih naloga je optimizacija poslovnih procesov v podjetju. Predstavlja platformo za organiziranje, avtomatizacijo in analizo metodologij, metrik, procesov in sistemov, ki merijo zmogljivost poslovanja. Običajen proces upravljanja poslovnih zmogljivosti sestoji iz treh korakov (Blansfield, 2007):

- **Planiranja** – izvaja se ga z izdelavo planov, simulacij in scenarijev.
- **Nadzorovanja** – glavni pripomočki tega koraka so sistemi kazalnikov, poročila in alarmi.
- **Izvajanja in prilagajanja** – v pomoč služijo orodja za napovedovanje in modeliranje.

Značilnost BPM sistemov je, da se z združevanjem podatkov iz različnih virov, poizvedovanjem in analizo nad njimi, pridobi informacije, ki so uporabne pri optimizaciji procesov. Gre za princip povratne zanke. S pomočjo BPM je tako možno zaznati in odstraniti probleme v začetnih fazah, še preden se ti razrastejo do te mere, da bi lahko ogrozili dosego zastavljenih ciljev podjetja. V ta namen se pri BPM uporablja sistem uravnoteženih kazalnikov (angl. *Key Performance Indicators, KPI*).

Sistem sestavlja skupina finančnih in nefinančnih kazalnikov, ki skupaj kot celota predstavljajo trenutno »zdravje« podjetja ter obenem nakazujejo smer gibanja. Izbor kazalnikov je zahtevna naloga. Obstaja sedem lastnosti, ki naj bi jih imel vsak kazalnik (TDWI, 2007):

- **Standardiziranost** – mere, ki jih kazalniki vsebujejo, morajo biti splošno sprejete znotraj organizacije.
- **Pravilnost** – za vsak kazalnik mora obstajati način, s katerim se ga preveri s podatki v transakcijskem sistemu. Navzkrižna kontrola povečuje zaupanje v vrednosti pri končnih uporabnikih.
- **Razumljivost** – kazalnik mora biti lahko razumljiv uporabnikom.
- **Soodvisnost** – kazalniki ocenjujejo uspešnost glede na pričakovanja. Pričakovanja so lahko:
    - meje, v katerih so vrednosti kazalnikov sprejemljive,
    - vnaprej postavljeni cilji (npr. 10% povečanje kupcev na četrtletje) ter
    - vrednost, ki so pričakovane za posamezno panogo.
- **Aktivnost** – kazalnik mora predstavljati akcijo (npr. prodaja na zaposlenega).
- **Spodbujevalnost** – kazalnik mora s svojo vsebino spodbujati k proaktivnosti zaposlenih (npr. namesto padcev se meri rast).
- **Stalnost** – kazalnik mora biti nespremenjen čez daljše časovno obdobje.

Sistem uravnoteženih kazalnikov je uporabnikom običajno na voljo preko nadzornih plošč, dosegljivih na portalih podjetja. Če kazalniki ustrezajo omenjenim lastnostim, pridobijo na priljubljenosti in s tem postanejo uporabni pri vsakdanjem upravljanju podjetja.

3.1.2 Faze evolucije podatkovnega skladiščenja

Management uspešnosti in učinkovitosti je neločljivo povezan s temelji poslovne inteligence in podatkovnega skladiščenja. Podatkovna skladišča integrirajo podatke iz različnih virov z namenom izdelave različnih kazalnikov, poročil, nadzornih plošč in planov. Odvisno od stopnje zahtev, ki jih okolje podatkovnega skladiščenja rešuje, ločujemo pet faz podatkovnega skladiščenja. Vprašanja posameznih faz so (Brobst, 2001):

1. Kaj se je zgodilo?
2. Zakaj se je zgodilo?
3. Kaj se bo zgodilo?
4. Kaj se trenutno dogaja?
5. Kaj želimo, da bi se zgodilo?

Za vsako od faz je značilno, da predstavlja nadgradnjo prejšnje faze. Poglejmo jih malo podrobneje.

Faza 1: »Poročila« – Obdobje statičnih poizvedb

Za prvo fazo podatkovnega skladiščenja je značilno, da je večina energije vložena v izgradnjo osrednje baze podatkov. Transakcijski podatki se iz različnih koncev podjetja integrirajo v enotni podatkovni repozitorij. S tem je omogočeno izvajanje odločitev, ki povezujejo različna funkcionalna področja znotraj podjetja.

Poizvedbe nad podatki so relativno enostavne, saj se večinoma izvajajo v obliki preddefiniranih poročil po določenem urniku v obliki vnaprej planiranih obdelav. Uporabniki podatkovnega skladišča v tej fazi večinoma samo izvajajo poročila, ki so jih po predhodnem dogovoru naredili informatiki. Funkcionalnosti poročil so omejene na parametrsko poizvedovanje in dimenzionalno analizo. Poročila so v večini primerov izdelana z uporabo poizvedb SQL ob izdatni pomoči administratorjev baz podatkov. Ti poizkušajo s pomočjo uporabe indeksov in agregatnih tabel reševati vsak problem posebej.

Faza 2: »Analiza« – Obdobje dinamičnih poizvedb

Uporabnike bolj kot kaj se je zgodilo zanima, zakaj se je nekaj zgodilo. V tej fazi se večina aktivnosti, ki je bila prej na strani informatikov, prenese na uporabnike podatkovnega skladišča. To so ljudje, ki se spoznajo na poslovanje podjetja, običajno pa nimajo pa znanja jezika SQL. Uspešen prehod v drugo fazo zato potrebuje uporabo grafičnih vmesnikov. To so poslovne rešitve (npr. Business Objects, Cognos, Micro Strategy), ki s pomočjo semantične plasti preslikajo fizično strukturo baze podatkov v uporabnikom razumljivejšo poslovno strukturo. V okolju dinamičnih poizvedb se vprašanja rojevajo dnevno. To pa na strani SUBP pomeni

nepredvidljive poizvedbe. Nujno je, da ima SUBP vgrajeno optimizacijo poizvedb glede na njihovo zahtevnost (ang. *Cost Based Optimization*), saj izdelava namenskih indeksov in agregatnih tabel zaradi nepredvidljivosti uporabniških poizvedb v tem primeru ne zadostuje več.

Faza 3: »Napovedovanje« – Uporaba analitičnih modelov

Podjetje je v tej fazi že osvojilo odgovore na vprašanja »kaj« in »zakaj« se je nekaj zgodilo. Glavna značilnost tretje faze je, da podjetje poizkuša napovedovati, kaj se bo zgodilo. Ta faza poizkuša s pomočjo uporabe podatkovnega rudarjenja napovedovati bodoče dogodke.

Za metode podatkovnega rudarjenja je značilno izvajanje zahtevnih matematičnih operacij in algoritmov. Običajni uporabniki podatkovnega rudarjenja so običajno majhne skupine analitikov z močnim statističnim ali/in trženjskim znanjem. To majhno število uporabnikov lahko z izvajanjem zahtevnih operacij močno vpliva na delovanje sistema. V tej fazi srečamo dve pomembni zahtevi:

- prisotnost atomarnih podatkov v podatkovnem skladišču ter
- zmogljivost računalniških sistemov z vidika izvajanja matematičnih operacij.

Faza 4: »Operacionalizacija« – Nenehno polnjenje podatkovnega skladišča

Četrta faza predstavlja aktivno podatkovno skladišče. Za to fazo je značilno nenehno polnjenje podatkov v podatkovno skladišče s ciljem, da se čim bolj zmanjša časovni zamik, ko je podatek iz transakcijskega sistema dostopen v podatkovnem skladišču. Faza predstavlja tudi pričetek uporabe podatkovnega skladišča za množico dnevnih operativnih odločitev, kar pomeni povečanje zahtev po hitrih poizvedbah.

Faza 5: »Aktivacija« – Avtomatsko izvajanje odločitev

Predstavlja zadnjo fazo v evoluciji okolja za podatkovno skladiščenje. Izvajanje individualnih odločitev se v tej fazi prevesi v izvajanje odločitvene politike. Cilj te faze je, da se človeka kot vmesni člen pri odločanju izloči povsod tam, kjer ta ni več nujno potreben. Operativne odločitve se v tej fazi sprejemajo avtomatsko s pomočjo izvajanja odločitvenih pravil nad podatkovnim skladiščem.

3.1.3 Opredelitev aktivnega podatkovnega skladišča

Aktivno podatkovno skladiščenje je proces, ki ni omejen na samo tehnologijo. Izraz »aktivno« je najprej uporabilo podjetje Teradata, ko je hotelo poudariti eno izmed značilnosti aktivnega podatkovnega skladiščenja, to je procesiranje s

povratno zanko (angl. *Closed Loop Processing*). Podatkovno skladišče je aktivno, če zanj velja (Agosta, 2005):

- **Predstavlja eno verzijo resnice.** V praksi obstaja nevarnost, da podjetje podatke poleg v podatkovno skladišče shranjuje še v vzporedna področna skladišča. V primeru, da se kasneje podatki namesto iz podatkovnega skladišča črpajo iz področnih skladišč, ne moremo govoriti o eni verziji resnice in s tem ne o aktivnem podatkovnem skladišču.
- **Podpira različne tipe odločitev.** Poizvedbe v aktivnem podatkovnem skladišču podpirajo tako strateške in taktične kot tudi operativne odločitve. Ker se podatki polnijo kontinuirano, mora SUBP reševati težave, ki se pojavljajo s sočasnim dostopom različnih poizvedb. V primeru, da so podprte samo operativne poizvedbe, takšno podatkovno skladišče ni aktivno.
- **Upravlja s transakcijskim sistemom.** Aktivno podatkovno skladišče nadzoruje odločanje v transakcijskih sistemih. Te odločitve se sprejemajo avtomatsko s pomočjo različnih mehanizmov, kot so prožilci (angl. *triggers*), sporočilnimi mehanizmi in namenskimi programskimi vmesniki. Če prenos informacij iz podatkovnega skladišča v transakcijski sistem ni avtomatski, velja, da podatkovno skladišče ni aktivno.
- **Predstavlja sistem s povratno zanko.** V praksi to pomeni, da je podatkovno skladišče uporabljeno kot vir informacij, ki optimizirajo transakcijski sistem. Transakcijski sistem polni podatkovno skladišče, ki pa v nasprotni smeri z informacijami pridobljenimi s pomočjo analiz vpliva na operativne procese in s tem na podatke, ki jih shranjuje transakcijski sistem. Podatkovno skladišče na ta način zagotavlja operativno inteligenco.

3.2 TOPOLOGIJA PODATKOVNEGA SKLADIŠČENJA

V podatkovnem skladiščenju se danes uporablja več različnih topologij. Tabela 2 prikazuje rezultat raziskave podjetja Forrester, ki je leta 2004 ugotovilo, da je najpogosteje uporabljena topologija podatkovnega skladiščenja zasnovana v obliki centraliziranega podatkovnega skladišča, opcijsko razširjenega s področnimi podatkovnimi skladišči in operativnimi podatkovnimi shrambami.

V tabeli 2 se poleg centralnega podatkovnega skladišča, ki ga opisuje poglavje 2.3.1, omenja tudi nekatere druge oblike skladiščenja podatkov, kot sta področno podatkovno skladišče (angl. D*ata Mart*) in operativna podatkovna shramba (angl. *Operational Data Store, ODS*).

Tabela 2: Kakšna je vaša topologija podatkovnega skladiščenja?

Topologija	Število v %
Centralizirano podatkovno skladišče s področnimi podatkovnimi skladišči in operativnimi podatkovnimi shrambami	44,33
Nepovezana področna podatkovna skladišča	19,70
Centralizirano podatkovno skladišče	18,72
Povezana področna podatkovna skladišča	16,26
Navidezno podatkovno skladišče	0,99

Vir: Prirejeno po Agosta, 2004

Področno podatkovno skladišče

Glavna razlika med področnim in centralnim podatkovnim skladiščem je v tem, da je izgradnja področnega povezana s poslovnimi potrebami določenega oddelka v podjetju (Inmon, 1999). Vsak oddelek ima lahko svoje področno podatkovno skladišče. Izraz svoje ne predstavlja samo logične, ampak tudi fizično ločenost, kar pomeni svojo strojno opremo, svoje uporabniške programe, svoj sistem za upravljanje baz podatkov. Vsak oddelek si po svoje predstavlja, kako naj njihovo področno skladišče izgleda, kakšne podatke naj vsebuje, kako dolgo naj bodo podatki na voljo za analize, kakšna naj bo podrobnost podatkov. Obstajata dva tipa področnih podatkovnih skladišč:

- **Odvisno področno skladišče** je tisto, ki se polni iz enotnega vira podatkov – centralnega podatkovnega skladišča.
- **Neodvisno področno skladišče** je tisto, ki podatke za polnjenje jemlje iz transakcijskih sistemov v podjetju.

Operativna podatkovna shramba

Operativna podatkovna shramba je integrirana zbirka podatkov, ki je organizirana po poslovnih področjih. Od podatkovnega skladišča pa se razlikuje v tem, da vsebuje samo trenutne in ne zgodovinskih podatkov, podatki so sveži in detajlni. Operativne podatkovne shrambe se polnijo s strani transakcijskih sistemov. Opravljajo nalogo integratorja podatkov v operativnem okolju podjetja. Ker vsebujejo sveže podatke, se jih lahko uporablja pri kritičnih operativnih procesih v podjetju. Literatura (Inmon, 1995) deli operativne podatkovne shrambe glede na zaostanek podatkov v primerjavi s transakcijskimi sistemi na naslednje tri razrede:

- razred I – zaostanek do treh sekund,
- razred II – zaostanek do 2 uri,

- razred III – zaostanek do 24 ur.

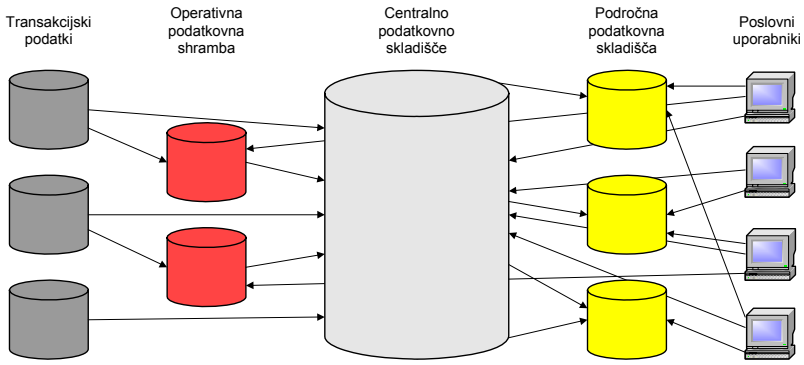
Najpogosteje se uporabljajo operativne podatkovne shrambe, ki so predstavniki razreda II ali III. Razred I se poredko uporablja iz vsaj dveh razlogov. Prvi je majhno število poslovnih procesov, ki bi potrebovali tako ažurnost podatkov, drugi pa je ekonomske narave, saj visoki stroški implementacije takšnega sistema težje upravičijo smiselnost naložbe.

Dolgoročno gledano bo po mojem mnenju uvajanje aktivnih podatkovnih skladišč vplivalo na zmanjševanje števila operativnih podatkovnih shramb. Aktivna podatkovna skladišča so centralizirana, po svoji definiciji pa naj bi zagotavljala sveže podatke in hitre odzivne čase pri izvajanju poizvedb. Ta dva kriterija, ki sta bila glavna razloga za nastanek in obstoj operativnih podatkovnih shramb, bosta tako izpolnjena drugje.

3.2.1 Topologija tradicionalnega podatkovnega skladiščenja

Topologijo tradicionalnega podatkovnega skladiščenja navadno sestavljajo operativne podatkovne shrambe, centralno podatkovno skladišče in odvisna področna skladišča podatkov. Predstavlja najpogostejši pristop v podatkovnem skladiščenju, kar je razvidno tudi iz tabele 2 na strani 26. Slika 4 prikazuje običajno topologijo tradicionalnega podatkovnega skladišča, ki pa se lahko v podrobnostih razlikuje glede na implementacijo (Theodoratos, 1999)

Slika 4: Topologija tradicionalnega podatkovnega skladišča



Vir: Prirejeno po Basu, 2003

Podatki iz transakcijskih sistemov se preko procesov za polnjenje podatkovnega skladišča (angl. *Extract Transform Load, ETL*), običajno v nočnem obdobju, prenašajo v dva sistema. Prvega predstavljajo operativne podatkovne shrambe, kjer so podatki na voljo operativnim poizvedbam. Drugi sistem pa predstavlja centralno podatkovno skladišče podjetja. Iz njega se lahko glede na potrebe

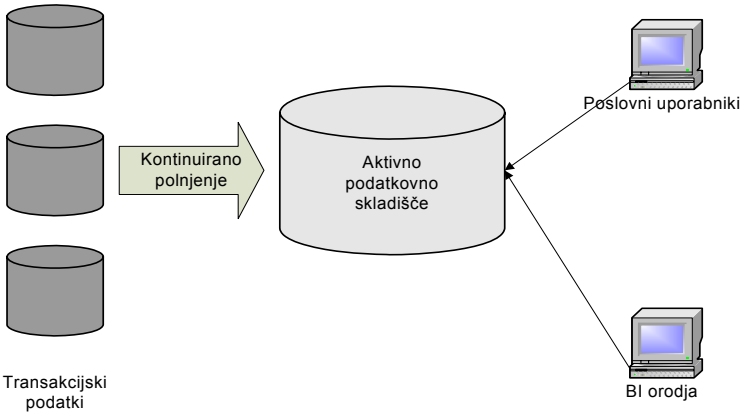
polnijo tudi ločena odvisna področna skladišča, ki zagotavljajo podatke posameznim oddelkom v podjetju.

3.2.2 Topologija aktivnega podatkovnega skladišča

Glavna značilnost topologije, ki podpira aktivno podatkovno skladiščenje, je fizično enotno skladišče podatkov. Sistem ni sestavljen iz ločenih zbirk podatkov, ki so značilne za tradicionalno podatkovno skladiščenje. Poizvedbe se sedaj ne glede svoj namen (strateške, taktične, operativne) izvajajo nad istim virom podatkov.

Pomembna razlika med tradicionalnim in aktivnim podatkovnim skladiščem se nahaja v implementaciji procesa polnjenja (slika 5). Ker se od aktivnega podatkovnega skladišča pričakuje pravočasno zagotavljanje informacij, polnjenje v obliki nočnih obdelav ne zadošča več. Zato se to izvaja med ostalimi procesi, tekom celotnega dneva. Govorimo o tako imenovanem kontinuiranem polnjenju.

Slika 5: Topologija aktivnega podatkovnega skladišča



Vir: Prirejeno po Basu, 2003

3.3 NAČINI POLNJENJA PODATKOVNIH SKLADIŠČ

V zgodovini se je razvilo več načinov polnjenja podatkov v podatkovna skladišča. Načini polnjenja podatkovnega skladišča se glede na čas nastanka delijo na:

- uporabo skript,
- uporabo orodij ETL,
- uporabo programov za integracijo aplikacij (angl. *Enterprise Application Integration, EAI*),

- uporabo programov za kontinuirano polnjenje (angl. *Transaction Data Movement, TDM*).

Izbira določenega načina polnjenja je odvisna od poslovnih zahtev, ki naj bi jih podatkovno skladišče izpolnjevalo. Poslovne zahteve določajo vrednosti kriterijem, ki morajo biti izpolnjeni. Najpomembnejši izmed njih so (Puttanqunta, 2006):

- obseg podatkov,
- pogostost polnjenja podatkovnega skladišča,
- sprejemljiva zakasnitev podatkov v podatkovnem skladišču,
- zagotavljanje integritete podatkov,
- omogočanje izvajanja transformacij,
- zahtevnost procesiranja ter
- vzdrževanje sistema.

3.3.1 Skripte

Skripte predstavljajo način polnjenja podatkov, ki se ga da najhitreje implementirati. Skripte so lahko razvite v kateremkoli programskem jeziku, ki podpira komunikacijo z bazo. To predstavlja fleksibilnost pri razvoju. Dejstvo, da so pisane v standardiziranih in razširjenih jezikih, prinaša pozitivne ekonomske učinke, ker se s tem izognemo visokim začetnim in vsakdanjim operativnim stroškom. Skripte se prožijo intervalno, običajno preko urnikov izvajanja (angl. *scheduled jobs*), ki so na voljo v vsakem operacijskem sistemu. Problem zna predstavljati njihovo vzdrževanje in administriranje. Izkaže se, da možnost hitrega razvijanja skript pogosto vodi v nesistematičnost in nedokumentiranost okolja, kar lahko na dolgi rok deluje kot sidro razvoja in obremenjuje razvijalce pri ostalem delu.

3.3.2 Orodja ETL

Orodja ETL odpravljajo nesistematičnost in nedokumentiranost skript. Omogočajo izdelavo procesov ETL, ki skrbijo za periodično polnjenje podatkovnega skladišča. Vsak proces ETL je sestavljen iz:

- Zajema (angl. *extract*) izvornih podatkov. Običajno je, da podatkovno skladišče vsebuje podatke, pridobljene iz različnih virov. Orodja ETL vsebujejo standardizirane vmesnike, ki omogočajo zajem podatkov iz relacijskih baz, datotek v različnih formatih, nestrukturiranih virih in podobno. Z uporabo teh vmesnikov se odstrani možnost napak pri zajemu

podatkov ter standardizira podatke tako, da so pripravljeni za naslednjo stopnjo – transformacijo.

- Transformacij (angl. *transform*) podatkov, ki odpravljajo pomanjkljivosti zajetih podatkov. Naloga transformacije je poenotenje podatkov, ki se izvrši z izvajanjem različnih funkcij in pravil, kot so:
    - transformacija šifriranih polj (npr. M v »moški«, Ž v »ženska«, 0 v neznano),
    - definiranje privzetih vrednosti (npr. če je podatek prazen, se ga transformira v »ni podatka«),
    - izračun novih vrednosti (npr. skupaj = cena + davek),
    - prirejanje umetnih ključev za potrebe počasi spreminjajočih se dimenzij,
    - preverjanje referenčne integritete,
    - transponiranje (angl. *transpose*) in vrtenje podatkov (angl. *pivot*) (pretvorba vrstic v stolpce in obratno).
- Prenosa podatkov (angl. *load*) v podatkovno skladišče. Pri tem obstaja več pristopov, ki so odvisni od poslovnih potreb. Možno je prekrivanje obstoječih podatkov (zgodovina se izgublja) in dodajanje podatkov (zgodovina se ohranja).

V splošnem gledano predstavljajo orodja ETL idealno rešitev za polnjenje velikega števila podatkov. Periodičen način izvajanja pa pogosto predstavlja pomanjkljivost, saj povečuje zaostanek podatkov in s tem zmanjšuje podatkovno svežino.

3.3.3 Integracija aplikacij (EAI)

Integracija aplikacij predstavlja kombinacijo procesov, programov, standardov in strojne opreme, ki dva ali več informacijskih rešitev v organizaciji združi tako, da jih zunanji opazovalec vidi kot eno celoto (ITToolbox, 2006). Tehnično gledano je integracije aplikacij narejena kot informacijska platforma, ki skrbi za komunikacijo, izmenjavo podatkov in sporočil s pomočjo vnaprej dogovorjenih vmesnikov (White, 2006).

Integracija aplikacij se je na področju podatkovnega skladiščenja pričela množičneje uporabljati s pojavom aktivnih podatkovnih skladišč. Tu je postala njena glavna naloga skrb za obojestransko komuniciranje s transakcijskim sistemom kot:

- odjemalec podatkov iz transakcijskega sistema,
- vir podatkov za izvajanje operativnih odločitev.

Skupna lastnost integracije aplikacij je, da:

- omogoča kontinuiran tok podatkov med izvornimi in ponornimi sistemi,
- zagotavlja uspešnost prenosa podatkov,
- podpira izgradnjo naprednih delovnih tokov (angl. *workflow*) ter
- omogoča osnovne transformacije.

Integracija aplikacij je v osnovi namenjena povezovanju aplikacij in ne podatkov. Zato je potrebno za začetna polnjenja podatkovnih skladišč uporabiti drug pristop. Ker gre za enkraten dogodek, se v ta namen najpogosteje uporabljajo skripte, možna pa je tudi uporaba orodij ETL.

Med delovanjem rešitve za integracijo aplikacij se podatki med sistemi prenašajo v majhnih količinah, v mnogokratnikih poslovnih dohodkov (en ali več). Integracija aplikacij se glede na način delovanja medsebojno razlikuje in deli na rešitve za (Gold-Bernstein, 1999):

- povezovanje platform,
- povezovanje podatkov,
- povezovanje poslovnih komponent,
- medaplikacijsko povezovanje,
- povezovanje procesov,
- povezovanje podjetij.

Za potrebe aktivnega podatkovnega skladiščenja so zanimive aplikacije za povezovanje podatkov, platform in procesov.

Povezovanje podatkov

Aplikacije za povezovanje podatkov predstavljajo najbolj razširjeno skupino rešitev za integracijo aplikacij. Sem spadajo različna orodja, ki delujejo po načelu ETL. Dobra lastnost teh informacijskih rešitev je ta, da implementacija običajno ne potrebuje spremembe na izvornem delu – viru podatkov.

Povezovanje platform

Povezovanje platform omogoča povezljivost različne strojne opreme, operacijskih sistemov in aplikacij. Obstaja več načinov povezljivosti platform, običajno se uporablja povezljivost preko sporočil, protokolov ORB (angl. *Object Request Broker*) in RPC (ang. *Remote Procedure Call*) (Gold-Bernstein, 1999).

Vsak od teh pristopov potrebuje za svoje delo implementacijo komunikacijskih vmesnikov na oddajni in sprejemni strani, kar predstavlja pomanjkljivost. Dobra

lastnost pa je omogočanje hitrejšega prenosa podatkov, ki namesto v paketni obliki tu poteka v obliki kontinuiranega toka.

Povezovanje procesov

Aplikacije za povezovanje procesov predstavljajo najvišji nivo informacijskih rešitev za integracijo aplikacij. Upraviteljem procesov omogočajo, da opredelijo, nadzorujejo in spreminjajo poslovne procese preko grafičnega modelirnega okolja. V primeru spremembe poslovnega procesa se ta sprememba odraža v aplikaciji za povezovanje procesov.

3.3.4 Kontinuirano polnjenje

Kontinuirano polnjenje (angl. *Transactional Data Movement, TDM*) predstavlja najmlajši način polnjenja podatkovnega skladišča. Eno izmed implementacij kontinuiranega polnjenja predstavlja kratica CDC (angl. *Changed Data Capture*), ki pomeni identifikacijo, zajem in dostavo podatkov, ki so se zgodili v informacijskem sistemu organizacije. Spremenjene poslovne zahteve so glavni razlog za uvedbo tega načina polnjenja. Glavni razlogi so (GoldenGate, 2007):

- **Globalizacija poslovanja**. V preteklosti je bilo običajno, da je vsako noč obstajal vnaprej znan interval – časovno okno, ko se je ustavila uporaba transakcijskih sistemov. V tem času so se prožile skripte oziroma procesi napisani z orodji ETL, ki so polnili podatkovno skladišče. Globalizacija je z neprekinjenim poslovanjem 24 ur na dan, 365 dni v letu odstranila to časovno okno, ko transakcijski sistemi ne delujejo.
- **Potreba po svežih podatkih**. Vrednost informacije pada s časom. Odločevalci potrebujejo sveže informacije, tudi takšne, ki temeljijo na dogodkih, nastalih v istem dnevu, uri ali manj.
- **Rast količine podatkov**. Vsakodnevno se analizira čedalje več podatkov. Po drugi strani pa globalizacija poslovanja zožuje ali celo odstranjuje časovno okno, ki je potreba procesov ETL.
- **Optimizacija stroškov**. Velike količine podatkov povečujejo zahteve sistemov, ki izvajajo skripte ETL. Optimizacija stroškov pa na drugi strani zmanjšuje temu namenjena sredstva. Tako prihaja do nasprotij, ko procesi ETL zaradi premalo zmogljivih sistemov ne morejo več opravljati svojih nalog.

Kontinuirano polnjenje poizkuša reševati zgoraj naštete poslovne potrebe s tem, da omogoča:

- **Polnjenje podatkovnega skladišča brez časovnega okna**, kar v praksi pomeni polnjenje preko celega dneva, ne da bi se ob tem obremenjevalo operativno delovanje sistema.
- **Povečevanje svežine podatkov**. S kontinuiranim polnjenjem se povečuje svežina informacij v aktivnem podatkovnem skladišču, kar vpliva na kvaliteto odločevanja odločevalcev. To povečevanje temelji na identifikaciji samo tistih podatkov, ki so bili spremenjeni v opazovanem sistemu.
- **Zmanjševanje stroškov**. Kontinuirano polnjenje zmanjšuje potrebe po računalniški zmogljivosti. Procesne zahteve procesa ETL, ki mora napolniti podatkovno skladišče v npr. 1 uri, se tako razdelijo na celoten dan, kar v praksi pomeni, da delo prvovrstnega strežnika sedaj opravlja običajna delovna postaja.

Kontinuirano polnjenje sestavlja več medsebojno povezanih komponent (Ankorion, 2005):

- **Agenti za zajem podatkov**. Njihova naloga je identifikacija in zajem spremenjenih podatkov v opazovanih sistemih. Agenti so prilagojeni posameznem viru podatkov. Zajem se običajno vrši iz dnevnikov sprememb in z uporabo prožilcev.
- **Procesi za obdelavo podatkov**. Predstavljajo kritični del kontinuiranega polnjenja. Njihove glavne naloge obsegajo skrb za zajem samo relevantnih podatkov (filtriranje), generiranja sekvenčnega mehanizma (na podlagi časa, transakcij ...) za potrebe določanja meja posamezne operacije, transformiranje podatkov (kot pri orodjih ETL) in določanje življenjske dobe zajetih podatkov.
- **Mehanizem dostavljanja.** Ko so podatki enkrat obdelani, se jih dostavi naprej, običajno napredni obliki orodja ETL. V osnovi obstajata dva modela obdelave podatkov, preprostejši model vlečenja (angl. *pull*) in zahtevnejši model potiskanja (angl. *push*) podatkov.
    - Za **model zajemanja podatkov** je običajno, da naročnik podatkov (aktivno podatkovno skladišče) periodično bere zajete podatke. Perioda je odvisna od poslovnih potreb (npr. vsako uro, vsakih 5 minut ...). Še vedno gre za implementacijo v obliki obdelav (kot pri običajnih orodjih ETL), prednost pa se kaže v tem, da ni več potrebe po nočnem oknu polnjenja in večji svežini podatkov.
    - Zahtevnejši **model potiskanja podatkov** ima korenine v aplikacijah za integracijo podatkov. Model temelji na tem, da naročnik čaka na nov podatek. Ko je nov podatek obdelan s strani procesov za obdelavo, ga naročnik takoj sprejme in shrani v podatkovno skladišče. Ta model zagotavlja minimalno zakasnitev med

transakcijskim sistemom in podatkovnim skladiščem, kar je osnovna zahteva implementacije aktivnega podatkovnega skladišča za odločanje v realnem času.

Kako izbrati primeren način za polnjenje podatkovnega skladišča? Pri odgovoru na to vprašanje si lahko pomagamo s tabelo 3, ki v odvisnosti od vrednosti posameznih kriterijev predlaga najprimernejši način.

Tabela 3: Načini zajemanja podatkov v podatkovno skladišče

KRITERIJ	SKRIPTE	ETL	EAI	TDM
Obseg podatkov	srednji	zelo velik	majhen	velik
Pogostost polnjenja	v presledkih	v presledkih	kontinuirana	kontinuirana
Zakasnitev	srednja do velika	srednja do velika	majhna	majhna
Integriteta podatkov	ne	včasih	zagotovljena	zagotovljena
Omogočanje transformacij	srednje	napredno	osnovno	osnovno
Zahtevnost procesiranja	v presledkih, visoka	v presledkih, visoka	kontinuirana, zmerna	kontinuirana, majhna

Vir: Prirejeno po Akbay, 2006

3.4 TEHNOLOŠKE ZAHTEVE AKTIVNIH PODATKOVNIH SKLADIŠČ

Vpletenost operativne poslovne inteligence v poslovanje zahteva od aktivnega podatkovnega skladišča, kot osrednjega dela infrastrukture, povečane oziroma spremenjene tehnološke zahteve. Zahteve lahko združimo v tri glavne skupine (Brobst, 2006a):

- zagotavljanje zmogljivosti,
- zagotavljanje razpoložljivosti in zanesljivosti,
- podatkovna ažurnost.

3.4.1 Zagotavljanje zmogljivosti

Zmogljivost je verjetnost, da sistem uspešno opravi (izpolni) dano nalogo v mejah določenih specifikacij (IEEE,1990). Pri aktivnem podatkovnem skladiščenju se s problemom zagotavljanja zmogljivosti srečujemo na dveh področjih:

- dodeljevanje virov,
- uporaba paralelizma.

3.4.1.1 Načini dodeljevanja virov

Za običajno podatkovno skladišče je značilno, da običajno ne vsebuje politike dodeljevanja svojih virov posameznim uporabnikom. Uporabniki svoje poizvedbe enostavno poženejo nad podatkovnim skladiščem, optimizator poizvedb (angl. *query optimizer*) na SUBP enakomerno porazdeli razpoložljive vire, čas trajanja poizvedbe pa je odvisen od zasedenosti sistema.

Pri aktivnem podatkovnem skladišču je potrebno bolj precizno načrtovanje porabe virov. Vsakemu tipu poizvedb (strateška, taktična, operativna) je potrebno določiti maksimalno dovoljeno porabo sistemskih virov. Planiranje virov je lažje pri operativnih poizvedbah, saj se poizvedbe dogajajo v odvisnosti od poslovnih procesov. Ker je s poslovnimi procesi neposredno povezana učinkovitost poslovanja podjetja, je potrebno, da se operativnim poizvedbam omogoči, da se izvajajo s predvidljivo hitrostjo, ne glede na vzporedne taktične in strateške ad-hoc poizvedbe. Obstaja več načinov zagotavljanja virov (Brobst, 2006a):

- vzporedni viri podatkov,
- proaktivno upravljanje poizvedb,
- uporaba človeškega nadzornika,
- dinamično dodeljevanje virov.

Vzporedni viri podatkov

Za način vzporednega vira podatkov je značilno, da so isti podatki podvojeni v dveh sistemih. To je pristop tradicionalnega podatkovnega skladišča z operativnimi podatkovnimi shrambami, kjer podatkovno skladišče skrbi za izvajanje strateških, operativne shrambe pa za izvajanje taktičnih poizvedb. Slabost tega pristopa je v tem, da so isti podatki shranjeni dvakrat, kar pomeni povečanje stroškov. Delno se da pristop optimizirati s skupno shrambo podatkov, ločeni pa so procesni deli (računalniki).

Proaktivno upravljanje poizvedb

Osnovni princip proaktivnega upravljanja poizvedb je, da obstaja nadzorni program, ki na podlagi tipa poizvedbe določi prednostni vrstni red (npr. operativne pred strateškimi). V primeru prezasedenosti sistema nadzorni program začasno odloži izvajanje strateških poizvedb na čas, ko sistem ni več toliko obremenjen, ali pa prekine izvajanje, če se izkaže, da bo poizvedba vrnila količinsko preobsežen rezultat.

Nadzorni program je lahko implementiran v SUBP ali pa v orodjih za poizvedovanje. Prednost implementacije v SUBP je v tem, da so vse poizvedbe obravnavane po istem pravilu, ne glede na to, iz kakšnega poizvedovalnega orodja izhajajo.

Uporaba človeškega nadzornika

Gre za enostaven način dodeljevanja virov. Administrator podatkovnega skladišča nadzira vitalne parametre sistema, kot so zasedenost začasnega pomnilnika za sortiranje podatkov, porabo procesorske moči, zasedenost vhodno-izhodnih virov, zaklepanje tabel. V primeru, da določena poizvedba (običajno strateška) zasede preveč virov, administrator z ročnim posredovanjem prekine to poizvedbo. Slabost tega pristopa je, da je poizvedba do prekinitve že uporabljala vire, ki bi jih lahko namesto nje uporabljale druge poizvedbe, ki pa so se zato izvajale počasneje.

Dinamično dodeljevanje virov

Predstavlja najnaprednejši način dodeljevanja virov v aktivnem podatkovnem skladišču. Viri se dodeljujejo dinamično, glede na trenutno stanje poizvedb v podatkovnem skladišču. Prioriteto določene poizvedbe določa rezultat večkriterijske funkcije, ki jo običajno določajo kriteriji, kot so uporabnik, vrsta aplikacije, potrebe po virih, čas znotraj dneva ... Sestavni del dinamičnega dodeljevanja virov je prej omenjeno proaktivno upravljanje poizvedb, ki skrbi, da ne bi prišlo do preobremenitve sistema. V praksi se uporaba dinamičnega dodeljevanja virov kaže na več načinov, naj omenim nekaj izmed njih (Brobst, 2006a):

- skrb za 100% izkoriščenost sistema, vse dokler obstajajo poizvedbe,
- avtomatično zmanjševanje virov pri poizvedbah z manjšo prioriteto, če je med njihovim izvajanjem prišla v izvajanje poizvedba z večjo prioriteto, in
- dinamično prilagajanje virov med posameznimi deli poizvedb.

3.4.1.2 Zagotavljanje zmogljivosti na nivoju fizičnega podatkovnega modela

Način dostopa do podatkov medsebojno razlikuje operativne in strateške poizvedbe. Operativne poizvedbe običajno dostopajo do podrobnih podatkov, v relativno omejenem obsegu. S stališča SUBP je pri takšnem dostopu najlažje povečati zmogljivost sistema z uporabo indeksov. Pri tradicionalnem podatkovnem skladiščenju, kjer strateške poizvedbe dostopajo do večje količine podatkov, se indeksi ne izkažejo kot najboljša izbira. Glavna načina za povečanje zmogljivosti SUBP na nivoju fizičnega modeliranja sta uporaba agregatnih tabel in branje celotnih tabel (angl. *full table scan*).

Velikosti blokov na disku

Na nivoju SUBP je hitrost bralnih operacij odvisna od fizične velikosti blokov na disku, kjer so shranjeni podatki. En blok predstavlja najmanjšo količino podatkov, ki jih računalnik prebere.

Za operativne poizvedbe, ki običajno dostopajo do podatkov preko indeksov, je bolje, da so podatki shranjeni v manjših blokih. Na ta način se zagotovi, da se z diskov ne bere preveč nepotrebnih podatkov. Nasproten primer pa predstavljajo strateške poizvedbe z branjem velikih količin podatkov. Ker bi pogosto branje manjših blokov vodilo v zmanjševanje hitrosti, je priporočljivo, da se za shranjevanje uporablja čim večje bloke. Večja velikost bloka se kaže kot povečanje hitrosti poizvedb. Vzrok je v tem, da diskovnim glavam pri večjih blokih ni potrebno tako pogosto spremeniti svoje pozicije, kot bi jo morali, če bi bili bloki manjši.

Opisano razlikovanje velikosti blokov predstavlja problem pri aktivnem podatkovnem skladiščenju. Zaradi kombinacije operativnih in strateških poizvedb nad istimi podatki prej opisani pristop ne zagotovi optimalne zmogljivosti sistema. Med proizvajalci SUBP je do sedaj problematiko najuspešneje rešilo podjetje Teradata (Brobst, 2006a). Namesto uporabe velikih blokov so na voljo cilindri. Značilnost cilindrov je, da jih sestavljajo manjši bloki. V primeru operativnih poizvedb se berejo bloki, če pa optimizator poizvedbe ugotovi, da gre za strateško poizvedbo, se berejo cilindri.

3.4.1.3 Uporaba paralelizma

Podatkovno skladiščenje uporablja za svoje delovanje nadpovprečno zmogljive sisteme. Običajni enoprocesorski računalniki ne zagotavljajo zadostne zmogljivosti sistema, saj ne zagotavljajo paralelizma. Zato se na področju aktivnega podatkovnega skladiščenja uporabljata dve konfiguraciji sistemov:

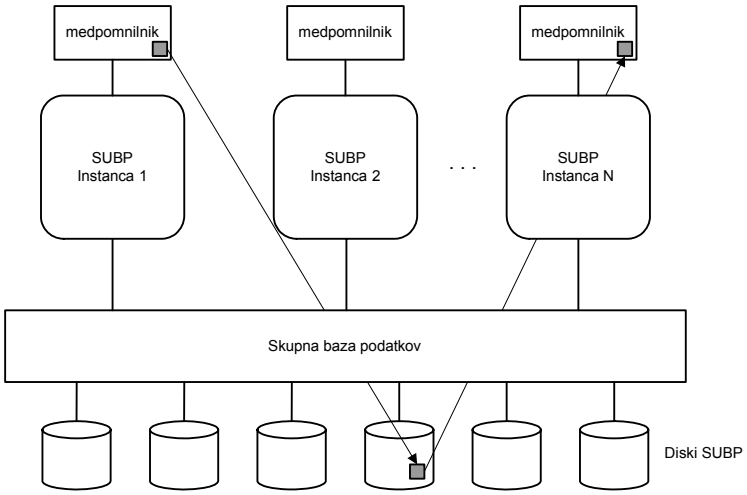
- gruča večprocesorskih računalnikov (angl. *Symmetric Multi Processing, SMP*),
- paralelni računalniki (angl. Massively *Parallel Processing, MPP*).

Za obe vrsti sistemov je značilna prisotnost paralelizma. Znotraj SUBP se paralelizem kaže kot povečanje števila poizvedb, ki se lahko izvajajo istočasno, in kot delitev posamezne poizvedbe na več korakov, ki se izvajajo vzporedno. V poglavju 3.4.1.2 omenjena lastnost uporabe indeksov in dostopa do podrobnih podatkov vodi v težave pri obeh implementacijah sistemov. Ker te težave lahko močno vplivajo na zmogljivost in razširljivost aktivnega podatkovnega skladišča, jim bom opisal podrobneje.

Uporaba gruče večprocesorskih računalnikov

Za to konfiguracijo je značilno, da več instanc SUBP dostopa do skupne baze podatkov, kar pomeni, da ne glede na to, na kateri instanci se izvaja poizvedba, vidi ta iste podatke.

Slika 6: Prikaz osveževanja podatkov pri gruči večprocesorskih računalnikov



Vir: Avtorjevo lastno delo

Tradicionalno podatkovno skladišče se običajno polni v nočnem času, ko ni uporabniških poizvedb; uporabniške poizvedbe pa se izvajajo v dnevnem času, ko ni spreminjanja podatkov. Vsak strežnik (v nadaljevanju instanca) ima na voljo medpomnilnik (angl. *buffer*), ki poveča hitrost poizvedb. Ta je transakcijsko sinhroniziran z bazo podatkov. Po opravljenem paketnem polnjenju, ki se običajno zgodi v eni transakciji, SUBP na vsaki instanci poskrbi za sinhronizacijo spremenjenih blokov v medpomnilniku.

Pri aktivnem podatkovnem skladišču se polnjenje ne dogaja samo enkrat v obliki nočnih paketnih obdelav, temveč kontinuirano preko celotnega dne. Medtem, ko ena instanca polni aktivno podatkovno skladišče, lahko druga dostopa do podatkov, ki so bili pravkar shranjeni. Prej omenjeni mehanizem transakcijskega sinhroniziranja baze podatkov in medpomnilnikov v tem primeru povzroči povečanje vhodno-izhodne komunikacije, kar je vidno na sliki 6. S povečevanjem števila instanc, s čimer se običajno povečuje zmogljivost in razpoložljivost sistema, se komunikacija samo še povečuje, kar posledično lahko pripelje do zmanjšanja zmogljivosti sistema.

Opisani problem je možno omejiti z lokalizacijo podatkov tako, da so instance specializirane. Specializacija pomeni delitev dela glede na določen kriterij, npr. na posamezna poslovna področja (finance, prodaja ...) ali na posamezna obdobja

(arhiva, letošnje leto). Na ta način se da doseči to, da posamezne instance dostopajo samo do določenega dela podatkov v skupi bazi podatkov. S tem se prepreči oziroma vsaj zmanjša možnost, da več instanc sočasno dostopa do istih podatkov.

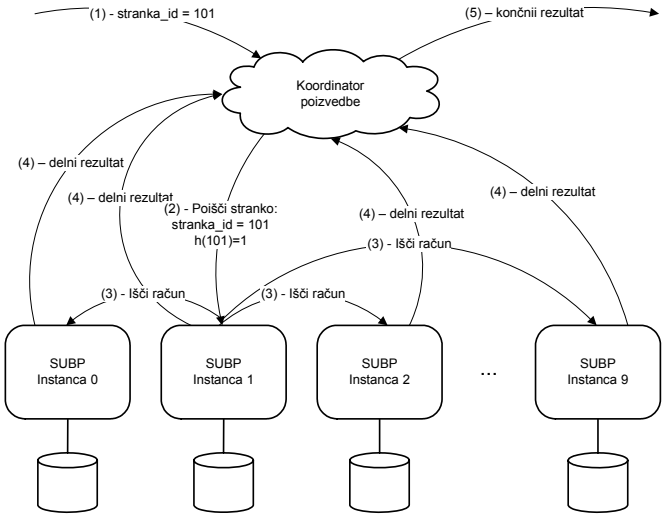
Uporaba pristopa paralelnih računalnikov

Glavna značilnost uporabe paralelnih računalnikov je masovno paralelno procesiranje. Velike tabele so pri tem pristopu razdeljene čez vse instance, vendar vsaka instanca SUBP dostopa samo do svojih podatkovnih diskov. V medpomnilniku vsake instance se nahajajo bloki, ki jih lahko spreminja samo ista instanca, zato ne prihaja do prej omenjenega problema sinhronizacije spremenjenih blokov. Čeprav izgleda, da gre za popolno rešitev, se pri izvajanju operativnih poizvedb lahko pojavi težava, ki pa ni prisotna pri strateških poizvedbah. Na primeru stavka SQL:

```
SELECT s.priimek, s.prebivalisce, r.stevilka,   , r.znesek
FROM   stranka s, racuni r
WHERE  s.stranka_id = r.stranka_id AND s.stranka_id = :par
```

predpostavljamo, da so podatki obeh tabel porazdeljeni po desetih instancah z uporabo funkcije hash h(). V primeru poizvedbe za šifro stranke 101 (1), se podatki o stranki preberejo z instance 1 (2). Ker je naslednji korak poizvedbe iskanje računov za to stranko, tabela računov pa je porazdeljena, se izvede poizvedba po vseh desetih instancah (3). Nekatere instance lahko vrnejo eno ali več, večina pa nič zapisov (4). Koordinator poizvedbe delne rezultate združi in vrne končni rezultat (5).

Slika 7: Prikaz izvajanja problematične poizvedbe pri uporabi paralelnih računalnikov



Vir: Avtorjevo lastno delo

Na sliki 7 se vidi, da predstavlja problem nekontrolirano izvajanje koraka 3 čez vse instance. Če se je zgornji tip poizvedbe izvede poredko za potrebe strateškega odločanja, opisana problematika ni tako moteča, saj se kompenzira s hitrostjo strojne opreme. Problem pa obstaja pri ponavljajočih se operativnih poizvedbah. Stalno izvajanje poizvedb tega tipa lahko zaradi neprestanega obremenjevanja instanc pripelje do močnega zmanjšanja zmogljivosti sistema. Rešitev problema predstavlja uvedba globalnega indeksa, ki ima podatke indeksirane čez vse instance. Koordinator poizvedb lahko z njegovo uporabo natančno določi, katere instance imajo želene podatke in tako zmanjša število izvajanj koraka 3.

3.4.2 Zagotavljanje razpoložljivosti in zanesljivosti

3.4.2.1 Opredelitev pojmov

Sistem

Sistem je zbirka komponent in/ali podsistemov, ki so povezani skupaj z namenom, da opravljajo določeno nalogo v okviru dogovorjenih zmogljivosti in zanesljivosti. Vrsta komponent, njihovo število, njihova kakovost in način, kako so povezani med sabo, neposredno vplivajo na zanesljivost sistema (Weibull, 2007).

Razpoložljivost

Razpoložljivost je mera, ki pove kakšna je verjetnost, da bo sistem deloval določen čas. Razpoložljivost sistema (A) določa naslednja relacija:

$$A = \frac{t_D}{t_D + t_N}$$

pri čemer t_D pomeni čas, ko je bil sistem delujoč, in t_N čas, ko je bil sistem nedelujoč (IEEE,1990).

Zanesljivost

Zanesljivost je verjetnost, da bo sistem pri določenih pogojih določen čas zadovoljivo opravljal svojo nalogo (Weik, 1996).

Razliko med zanesljivostjo in razpoložljivostjo si lahko razložimo na praktičnem primeru:

- **Zanesljivost**: V času trajanja poleta z letalom želimo, da bo zanesljivost letala čim bližje 100 % (99,9999 %), ko pa je letalo parkirano, nas njegova zanesljivost ne skrbi več.
- **Razpoložljivost**: Za računalniški sistem želimo, da bo v enem letu deloval z 99,90% razpoložljivostjo, kar pomeni, da v povprečju 8,76 ure v letu ne bo deloval.

3.4.2.2 Reševanje problemov, povezanih z zagotavljanjem razpoložljivosti

Uporaba aktivnega podatkovnega skladišča za potrebe operativnega odločanja zahteva povečano razpoložljivost sistema. Ta je potrebna predvsem pri področjih, ki jih bom opisal v nadaljevanju.

Polnjenje podatkov

Tradicionalno podatkovno skladišče se običajno polni v nočnem času, ko se nad podatki ne izvajajo poizvedbe. Aktivno podatkovno skladišče se polni kontinuirano preko celotnega dne, v istem času, kot se izvajajo tudi poizvedbe nad podatki. Pri tem se srečujemo s problemom hitrega zagotavljanja pravilnih podatkov delujočim poizvedbam. Največkrat se problem pojavi takrat, ko želi poizvedba dostopati do podatkov, ki se nahajajo v tabelah, ki se medtem polnijo z novimi podatki. Čeprav SUBP omogočajo sočasne dostope tega tipa, pa se v določenih primerih lahko zgodi zaklepanje dela table. Najbolj problematično je to pri velikih tabelah, ki so v času polnjenja nedosegljive za poizvedbe. V tem primeru se zgodi, da mora poizvedba čakati, da se prejšnja operacija konča. Če je poizvedba operativnega tipa, se opisani problem odraža kot zastoj pri operativnem poslovanju podjetja, kar pa ni sprejemljivo. Problem je rešljiv na ta način, da se velike tabele fizično razbije na vsaj dve tabeli:

- prva, velika statična, vsebuje arhivske podatke, ki se ne spreminjajo (npr. vsa zgodovina vključno s včerajšnjim dnem),
- druga, manjša dinamična pa podatke, ki se spreminjajo (npr. za tekoči dan).

Obe tabeli se združi s pogledom (angl. *view*). Poizvedba nad pogledom se pri tem pristopu fizično razdeli na dve poizvedbi, ena nad veliko statično tabelo in druga nad manjšo dinamično tabelo. V primeru sočasnega polnjenja in poizvedbe se poizvedba nad dinamično tabelo za majhen čas upočasni ali ustavi, nad veliko statično pa se nemoteno odvija naprej. Ker je statična tabela lahko nekajkrat (tudi več 100-krat) večja od dinamične tabele, se ta upočasnitev ne opazi. Pristop ima eno pomanjkljivost, to je zagotovitev mehanizma, ki prenaša podatke iz dinamične v statično tabelo po tem, ko je za del podatkov znano, da se ne spreminjajo več.

Izdelovanje varnostnih kopij

Pri podatkovnem skladiščenju običajno v nočnem času obstaja prosti interval, v katerem lahko izvedemo varnostno kopiranje baze podatkov. Ker sistem v tem času ni obremenjen z uporabniškimi poizvedbami, se lahko uporabi strategija izdelovanja statičnih kopij (angl. *cold backup*). Zanjo je značilno, da se ustavi delovanje baze, izvede kopiranje datotek na datotečnem strežniku in po uspešnem kopiranju ponovno zažene sistem za upravljanje baze podatkov.

Aktivno podatkovno skladišče ne sme uporabljati te strategije. Pri delujočem sistemu je zato potrebno izvesti kopijo podatkov med delovanjem sistema (angl. *hot backup*). Izvajati ga je mogoče, če SUBP podpira mehanizem shranjevanja sprememb nad podatki v dnevnike sprememb (angl. *archive log*). Pri izdelavah varnostnih kopij se namesto velikih fizičnih datotek, ki predstavljajo bazo podatkov, shranjujejo samo dnevniki sprememb, ki so mnogo manjši, njihova vsebina pa zadošča za rekonstrukcijo podatkov v primeru napak. Uporaba dnevnikov prinaša tudi to prednost, da je v primeru napake na sistemu bazo podatkov možno rekonstruirati v stanje, ki se je zgodilo do zadnje potrjene transakcije, kar dodatno povečuje vrednost takšnega pristopa izdelave varnostnih kopij.

Skrbniške operacije na SUBP

Podatkovno skladišče potrebuje vsake toliko časa intervencijo s strani skrbnika baze. Nad SUBP se izvajajo operacije reorganizacij podatkov, nastavljanje parametrov delovanja in tudi indeksiranje tabel ter osveževanje statistik. Aktivno podatkovno skladišče mora omogočati, da je možno te, sicer nadzorovane, posege izvajati tako, da ne vplivajo na zmanjšanje razpoložljivost sistema. Najbolje se v tem primeru obnesejo sistemi, ki se jih uporablja za stalno delujoče operativne transakcijske baze podatkov.

Vzdrževanje in nadgrajevanje strojne opreme

Večina strežnikov ima danes vgrajene diagnostične mehanizme, ki napovedujejo možne strojne napake. Zato se določeni deli strojne opreme preventivno zamenjujejo, še preden do teh napak pride. Najpogosteje se posegi preventivnega zamenjevanja komponent izvajajo pri diskovnih kapacitetah, napajalnikih in kontrolerjih.

Spremembe na konfiguraciji pa niso vedno vzrok preventivnega odpravljanja napak, ampak tudi nadgrajevanja sistema zaradi povečanih potreb. V tem primeru se običajno povečujeta dva segmenta:

- Diskovne kapacitete. Potreba aktivnega podatkovnega skladišča po hranjenju atomarnih zgodovinskih podatkov ima za posledico stalno povečevanje diskovnih kapacitet.

- Procesni del. Rast uporabe aktivnega podatkovnega skladišča vpliva na potrebo po nadgrajevanju procesorjev in spomina. Odvisno od platforme se ta poseg izvaja z dodajanjem dodatnih procesorjev in pomnilniških vezij ter dodajanjem novih ali zamenjavo obstoječih procesnih enot (računalnikov).

SUBP mora omogočati, da se takšni posegi izvajajo brez vpliva na razpoložljivost sistema.

Nadgradnja programske opreme

Proizvajalci operacijskih sistemov in SUBP nenehno izpopolnjujejo svoje izdelke. Posodobitve so uporabnikom na voljo v obliki servisnih programskih paketov. Priporočljivo je, da se vse programske nadgradnje pri aktivnem podatkovnem skladiščenju izvajajo pri delujočem sistemu.

3.4.2.3 Izpadi sistemov

Ne glede na to, da je vsa pozornost usmerjena k preprečevanju izpadov sistema, pa včasih vseeno pride do izpadov sistemov. Govorimo o dveh tipih izpadov:

- načrtovanih izpadih sistemov,
- nenačrtovanih izpadih sistemov.

S strani zagotavljanja razpoložljivosti so problematični nenačrtovani izpadi sistemov. Ti se dogajajo kot posledica (Brobst, 2006):

- napak na strojni opremi,
- napak v programski opremi,
- katastrof in
- človeških napak.

Odpravljanje strojnih napak

Običajno je, da se napake na strojni opremi pojavljajo posamezno, zato zadostuje uvedba redundantnih komponent s ciljem preprečitve izpada. Rešitve vsebujejo:

- uvedbo diskovnega polja z vgrajenim sistemom varovanja podatkov RAID,
- uporabo podvojenih vhodno-izhodnih kontrolerjev, hladilnih enot, mrežnih poti,
- uporabo paritet pri spominskih kapacitetah, s čimer se prepreči zaradi napačnega bita,
- toleriranje manjših napak v procesorju.

Kratica RAID (angl. *Redundant Array of Independent Disks*) pomeni redundantno skupino neodvisnih diskov. Predstavlja najpogostejši način združevanja diskov v

eno logično enoto. RAID pozna več načinov oziroma nivojev povezovanja diskov, najzanimivejša za potrebe podatkovnega skladiščenja sta konfiguraciji RAID tipa 1 in 5. Prva je preprostejša in njena edina naloga je zrcaljenje podatkov med dvema diskoma. Če odpove en disk, so vsi podatki na drugem disku še vedno na varnem. Če je RAID 1 dokaj enostaven, je RAID 5 zapletenejši in za delovanje potrebuje vsaj tri diske. Na prvih dveh so zapisani podatki, na tretjem pa pariteta oziroma vsota po modulu dva prvih dveh diskov. Če odpove kateri od glavnih dveh diskov, so podatki še vedno na voljo, saj se jih da po isti operaciji preračunati nazaj za manjkajoči disk. Izkoriščenost diskov, varovanih s sistemom RAID, je odvisna od izbire tipa varovanja. RAID 1 omogoča izkoriščenost diskov 50 %, pri RAID 5 pa se izkoriščenost z vsakim dodatnim diskom boljša, pri treh je 66 %, pri štirih pa že 75 % (DBA Support, 2006).

Odpravljanje programskih in človeških napak

Drugi tip nenačrtovanih izpadov sistema je posledica napak na programski opremi. Te napake, za razliko od odpovedi posameznih komponent pri strojnih napakah, povzročijo izpad celotnih podsistemov podatkovnega skladišča. Običajno se zgodi, da odpovesta SUBP ali operacijski sistem, medtem ko fizična baza podatkov še vedno deluje. Izpad sistema se v teh primerih lahko zagotovi z uporabo paralelnih strežnikov. Na vsakem od teh strežnikih, ki so postavljeni v gruče, se na operacijskem sistemu izvaja ločen SUBP. Vsi strežniki imajo povezavo do skupne baze podatkov. V delujočem stanju strežniki enakomerno sodelujejo pri izvajanju poizvedb, s čimer se zagotavlja večja zmogljivost sistema. V primeru izpada enega od strežnikov preostali delujoči prevzamejo njegove naloge. Zaželeno je, da se ta prevzem opravi avtomatsko, brez posredovanja človeka. Posredovanje človeka namreč vpliva na zmanjšanje razpoložljivosti sistema.

Odpravljanje napak zaradi katastrof

Odpravljanje napak zaradi katastrof predstavlja najzahtevnejšo in najdražjo možno implementacijo zagotavljanja razpoložljivosti sistema. Glede na povzročitelja lahko katastrofe delimo v dve skupini:

- naravne, ki so posledica potresov, poplav, požarov ...
- človeške, ki so posledica sabotaž, terorističnih napadov ...

Oba tipa katastrof imata za posledico popolno uničenje infrastrukture, kar pri aktivnem podatkovnem skladiščenju lahko kritično vpliva na delovanje in obstoj podjetja kot celote. Rešitev tega problema je v podvajanju celotnega sistema. Stroški in zahtevnost implementacije so odvisni od poslovnih zahtev. V vsakem primeru gre pri sistemih, ki so varni pred izpadi (angl. *fault tolerant*), za več kot podvojene stroške implementacije. Na nivoju infrastrukture je potrebno zagotoviti:

- podvojitev sistema, ki se fizično nahaja na geografsko oddaljeni lokaciji – najbolje na drugi tektonski plošči, s čimer se izolira možnost, da bi potres uničil oba centra,
- podvojitev omrežne infrastrukture za potrebe zajema in poizvedovanja podatkov,
- vpeljavo mehanizma, ki bo skrbel za sinhronizacijo podatkov v obeh sistemih.

Logistično najtežji primer predstavlja vpeljava mehanizma za sinhronizacijo podatkov, saj je njegova implementacija neposredno odvisna od poslovnih zahtev. Obstajata dva pristopa sinhronizacije podatkov:

- kopiranje podatkov na drug strežnik (angl. *replication*) in
- paralelno polnjenje podatkov.

Kopiranje podatkov predstavlja enostavnejšo implementacijo sinhronizacije. Izvaja se ga s pomočjo kopiranja baze podatkov in dnevnikov. Lastnost rešitve je, da povzroča časovni zamik med podatki v obeh sistemih.

Paralelno polnjenje podatkov predstavlja težjo implementacijo mehanizma sinhronizacije. Značilnost te implementacije je paralelno delovanje obeh sistemov (primarnega in rezervnega). Oba sistema skrbita za paralelno polnjenje podatkov iz transakcijskih sistemov, kar zagotavlja enakost podatkov do nivoja posamezne transakcije.

3.4.3 Podatkovna ažurnost

Operativna poslovna inteligenca potrebuje za svoje odločitve ažurne podatke v podatkovnem skladišču. Dobro načrtovano aktivno podatkovno skladišče lahko zagotovi prisotnost podatkov z minimalno zakasnitvijo glede na stanje v transakcijskih sistemih. Zagotavljanje takšne ažurnosti podatkov predstavlja povečanje stroškov, ki pa niso vedno poslovno upravičeni. Zato je potrebno, da se nivo ažurnosti podatkov v podatkovnem skladišču določa glede na poslovne potrebe.

3.4.3.1 Strategije polnjenja

Strategijo polnjenja podatkov v podatkovno skladišče lahko glede na hitrost polnjenja delimo na:

- popolno osveževanje,
- delno osveževanje in
- kontinuirano polnjenje.

Izbira ustrezne strategije je odvisna od tega, kakšen je cilj. Gre za izbiro med povečanjem svežine podatkov na eni strani in povečanjem hitrosti polnjenja.

Masovno procesiranje velike količine podatkov vodi v povečanje hitrosti polnjenja, toda akumuliranje podatkov za potrebe polnjenja prinaša na drugi strani povečanje časovnega zaostanka med podatkovnim skladiščem in transakcijskim sistemom.

Izbiro ustrezne strategije poleg poslovnih zahtev določajo tudi tehnični dejavniki kot so:

- potrebe po diskovnih kapacitetah,
- vpliv na izvajanje poizvedb,
- razmerje med količino novih in obstoječih podatkov,
- razmerje med hitrostjo ažuriranja zapisa in vstavljanja novega zapisa.

Popolno osveževanje

Popolno osveževanje predstavlja najenostavnejšo strategijo polnjenja. Njena značilnost je v tem, da se podatki v tabeli vsakokrat napolnijo v celoti. Koraki pri popolnem polnjenju se delijo na:

- netransakcijsko polnjenje podatkov (angl. bulk load),
- izdelavo indeksov,
- osveževanje statistik.

Strategija je primerna pri osveževanju majhnih tabel ali če je delež novih podatkov velik (10 % in več glede na velikost tabele) (Kimball, 1999). Pri tehnični implementaciji strategije popolnega osveževanja podatkov lahko pride do težav, če uporabniške poizvedbe dostopajo do podatkov v tabeli ravno v času njenega osveževanja. Rešitev tega problema je možna z uporabo začasnih tabel, kamor se podatki naložijo, indeksirajo in statistično ažurirajo. Ko je začasna tabela pripravljena za uporabo, se z uporabo pogleda (angl. *view*) izvede zamenjava produkcijske in začasne tabele. Novi podatki se tako v trenutku prikažejo uporabniku. Pomembno pri tem pristopu je zagotovitev, da se poizvedbe sklicujejo na logični pogled in ne na fizično tabelo, ter zagotovitev dodatnega diskovnega prostora za začasne tabele.

Slabost strategije popolnega osveževanja je v tem, da povzroča največji časovni zamik pri svežini podatkov. Problem se pojavi predvsem pri osveževanju velikih tabel, saj lahko prihaja pri tem do velikih časovnih zakasnitev med podatki v transakcijskem sistemu in aktivnem podatkovnem skladišču. Zato je strategija primerna le pri osveževanju manjših referenčnih tabel – šifrantov.

Delno osveževanje

Za strategijo delnega osveževanja je značilno polnjenje novih podatkov v obstoječo tabelo, ki že vsebuje stare podatke. Podatki se tako kot pri strategiji popolnega polnjenja v podatkovno skladišče polnijo v obliki netransakcijskega polnjenja. Delno osveževanje se da implementirati na dva načina:

- z direktnim polnjenjem v ciljno tabelo,
- z uporabo začasne tabele in kasnejšim polnjenjem v ciljno tabelo.

Izbira posamezne možnosti je odvisna od tega, kaj SUBP omogoča. Pri direktnem polnjenju v ciljno tabelo je pomembno, da SUBP:

- podpira avtomatsko osveževanje indeksov in statistik pri netransakcijskih operacijah,
- rešuje probleme, ki se pojavijo pri zaklepanju tabel in umazanem branju (angl. *dirty read*) zaradi netransakcijskega polnjenja podatkov.

V primeru, da SUBP ne podpira zgornjih zahtev, se delno osveževanje lahko implementira s pomočjo netransakcijskega polnjenja v začasno tabelo. V tem primeru je potreben dodaten prostor za začasno tabelo. Ko je začasna tabela naložena, se iz nje s pomočjo stavkov INSERT/SELECT ali MERGE SQL izvede transakcijsko polnjenje v končno tabelo. Slabost tega načina je v tem, da je zaradi transakcijskega polnjenja podatkov počasnejše, SUBP pa generira velike količine dnevnikov.

Kontinuirano polnjenje

Skupna značilnost predhodnih strategij je, da prihaja pri obeh zaradi polnjenja v obliki paketnih obdelav do zakasnitev in s tem do zmanjšanja svežine podatkov v podatkovnem skladišču. Strategija kontinuiranega polnjenja odpravlja to pomanjkljivost s stalnim polnjenjem na nivoju posameznih dogodkov. Podatkovno skladišče se pri tej strategiji polni s pomočjo uporabe stavkov INSERT in UPDATE SQL. Glavne razlike v primerjavi s paketnim polnjenjem so:

- Podatki iz transakcijskega sistema so lahko na voljo takoj, kar pri paketnih obdelavah ni možno.
- Zaradi načina polnjenja je poraba računalniških zmogljivost na nivoju posameznega zapisa mnogo večja kot pri paketnih obdelavah.
- Izvajanje stavkov SQL na nivoju posameznih transakcij omogoča izvajanje poizvedb med polnjenjem, kar pri netransakcijskem polnjenju ni možno.

Poraba računalniških zmogljivosti predstavlja največjo pomanjkljivost kontinuiranega polnjenja. Obstaja zelo majhna verjetnost, da bi poslovne zahteve zahtevale transakcijsko enakost svežine podatkov podatkovnega skladišča v

primerjavi s transakcijskim sistemom. Zato se v tem primeru lahko uporabi polnjenje z uporabo vmesnega pomnilnika. Temu se določi zgornja meja X (kapaciteta) in maksimalni čas čakanja Y. V primeru, da je katerikoli od parametrov X in Y presežen, se izvede transakcijsko polnjenje podatkovnega skladišča.

4 ANALIZA SISTEMA ZA UPRAVLJANJE BAZ PODATKOV ORACLE 10g

4.1 IZBIRA PRIMERA

V tretjem poglavju sem opisoval zahteve, ki so potrebne za implementacijo aktivnega podatkovnega skladišča. Ker sem želel preveriti, kako so te funkcionalnosti implementirane v praksi, bom v tem poglavju kot primer analiziral pripravljenost SUBP Oracle 10g za aktivno podatkovno skladiščenje. Za Oracle sem se odločil na podlagi najboljšega rezultata dveh kriterijev, ki sta ga predstavljali analizi podjetja Gartner. Analiza je ocenjevala:

- primernost SUBP za potrebe podatkovnega skladiščenja – za leto 2006,
- analizo prihodkov ponudnikov od prodaje SUBP – za leto 2005.

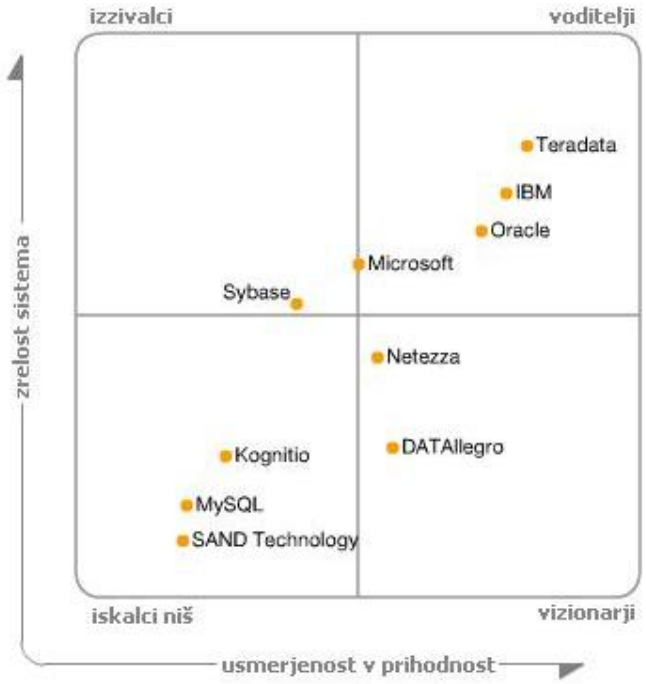
Magični kvadranti SUBP za potrebe podatkovnega skladiščenja – leto 2006

Prva analiza (Feinberg, 2006) je ocenjevala primernost različnih SUBP za potrebe podatkovnega skladiščenja. Kriterije za končno oceno so sestavljali:

- sposobnost kontinuiranega polnjenja podatkov,
- sposobnost izvajanja več tisoč standardnih operativnih poizvedb z uporabo indeksov,
- izvajanje naključnega števila ad-hoc strateških poizvedb nad naključnim naborom podatkov.

Analiza je ponudnike SUBP razvrstila v štiri skupine, ki jih prikazuje slika 8. Najboljši kandidati se nahajajo v zgornjem desnem kvadrantu (voditelji in vizionarji). Podjetja Teradata, IBM, Oracle in Microsoft, ki so se nahajala v tem kvadrantu, sem izbral v ožji izbor za naslednjo analizo.

Slika 8: Magični kvadranti SUBP za potrebe podatkovnega skladiščenja



Vir: Feinberg, september 2006

Analiza prihodkov od prodaje SUBP po posameznih dobaviteljih za leto 2005

Druga analiza podjetja Gartner (tabela 4) razvršča ponudnike glede na celotni prihodek, ki je bil ustvarjen s prodajo SUBP v letu 2005.

V analizi prihodkov se na prvih štirih mestih pojavljajo isti ponudniki kot v prejšnji analizi. Najbolj izstopa podjetje Oracle, saj njegov SUBP pokriva skoraj 50% tržni delež. Očitno odstopanje od ostalih SUBP mi je olajšalo izbiro SUBP Oracle za nadaljnjo analizo funkcionalnosti.

Tabela 4: Svetovnl prihodki od prodaje SUBP za leto 2005 (v milijonih $)

Podjetje	2005	2005 tržni delež v (%)	2004	2004 tržni delež v (%)	2004–2005 rast v (%)
Oracle	6.721,1	48,6	6.234,1	48.9	7,8
IBM	3.040,7	22,0	2.860,4	22.4	6,3
Microsoft	2.073,2	15,0	1.777,9	13.9	16,6
Teradata	440,7	3,2	412,1	3.2	6,9
Sybase	407,0	2,9	382,8	3.0	6,3
Ostali ponudniki	1.134,7	8,2	1.090,4	8.5	4,1
Skupaj	**13.817,4**	**100,00**	**12.757,8**	**100,00**	**8.3**

Vir: Gartner DataQuest, maj 2006

4.2 ZNAČILNOSTI SISTEMA ZA UPRAVLJANJE BAZ PODATKOV ORACLE 10g

4.2.1 Predstavitev podjetja Oracle

Začetki podjetja Oracle segajo v leto 1977, ko so trije ustanovitelji, Lawrence J. Ellison, Bob Miner in Ed Oates, ustanovili podjetje Software Development Laboratories. Cilj podjetja je bila komercialna implementacija SUBP, ki bi temeljil na relacijskem modelu in bi uporabljal poizvedovalni jezik SQL. Podjetje se je v treh desetletjih razvilo v giganta na področju podjetij za razvoj programske opreme. Konec leta 2006 je bilo glede na tržno kapitalizacijo drugo največje programsko podjetje na svetu z več kot 56.000 zaposlenimi po vsem svetu (Yahoo Finance, 2007). Čeprav je primarno področje Oracla še vedno razvoj, distribucija in vzdrževanje lastnih sistemov za upravljanje podatkovnih baz, se predvsem v zadnjih letih, tudi s pomočjo s pomočjo prevzemov, širi na ostala programska področja, ki podpirajo delovanje organizacij. Tako je močno prisoten na področju poslovnih aplikacij, ki podpirajo operativno delovanje posameznih panog, in poslovne inteligence.

4.2.2 Arhitektura SUBP Oracle 10g

Zaradi lažjega razumevanja, kaj Oracle sploh je, bom v tem poglavju opisal osnovne gradnike, ki sestavljajo SUBP Oracle 10g. Arhitekturo lahko delimo na:

- bazo podatkov (fizične in logične podatkovne strukture),
- strežnik (angl. *instance server*) (procesi in pomnilniška struktura).

Baza podatkov je prostor, kjer so shranjeni podatki. Podatki sami zase niso uporabni, če ne obstaja nek mehanizem, s pomočjo katerega se dostopa do njih. Za dostop do podatkov pri SUBP Oracle skrbi skupek procesov, ki dostopajo do Oraclu dodeljenega spomina. Strukturi procesov in dodeljenega spomina pravimo strežnik. Možno je, da eno bazo podatkov uporablja več strežnikov. Ta konfiguracija se v Oracle terminologiji imenuje RAC.

4.2.2.1 Fizična podatkovna struktura baze

Fizično strukturo baze sestavljajo datoteke na operacijskem sistemu. Glede na njihov pomen se delijo na:

- podatkovne datoteke (angl. *data files*),
- kontrolne datoteke (angl. *control files*),

- dnevnike za obnovo podatkov (angl. *redo log files*),
- arhivske dnevnike za obnovo podatkov,
- parametrsko datoteko,
- arhivske kopije datotek.

Podatkovne datoteke

Vsaka baza vsebuje eno ali več podatkovnih datotek, ki shranjujejo podatke. Za podatkovne datoteke je značilno, da:

- podatkovna datoteka pripada samo eni bazi podatkov,
- se vsaka podatkovna datoteka lahko poveča, če bazi zmanjka prostora za delo,
- ena ali več podatkovnih datotek sestavljalo logično enoto za shranjevanje podatkov, imenovano tablespace.

Podatki, shranjeni v podatkovnih datotekah, se berejo po potrebi. Prenašajo se v del pomnilnika računalnika, ki je dodeljen kot vmesni pomnilnik baze. Ko baza želi dostopati do nekega podatka, se najprej preveri, ali se ta podatek nahaja v vmesnem pomnilniku. Če se, se ga prebere od tam, drugače pa se ga prebere iz podatkovnih datotek. Na ta način se poveča hitrost dostopa do podatkov, saj se pogosto uporabljeni podatki (npr. majhni šifranti) z veliko verjetnostjo nahajajo v pomnilniku računalnika.

Vmesni pomnilnik pa je koristen tudi pri spreminjanju in dodajanju podatkov. Z namenom, da bi se zmanjšalo število dostopov do podatkovnih datotek in se na ta način povečalo hitrost, se spremenjeni podatki shranjujejo naenkrat v ciklih, ki jih določa proces za shranjevanje podatkov (DBWn). DBWn je en izmed procesov baze Oracle, ki bodo opisani v nadaljevanju tega poglavja.

Kontrolna datoteka

Kontrolna datoteka vsebuje podatke, ki opisujejo fizično strukturo baze podatkov. Med drugim vsebuje ime baze, lokacije ostalih datotek, podatke o kopijah, datum nastanka baze ... Ker predstavlja kazalo baze, je zelo priporočljivo, da obstaja več kopij kontrolne datoteke, ki se fizično nahajajo na različnih diskih. Oracle med delovanjem paralelno dostopa in spreminja vse kopije ter v primeru napake ene izmed njih to javi operacijskemu sistemu. Na ta način se zmanjša možnost izgube podatkov zaradi izgube kontrolne datoteke.

Dnevniki za obnovo podatkov

Vsaka baza vsebuje skupino dveh ali več datotek, katerih namen je shranjevanje podatkov, ki so bili spremenjeni, dodani ali brisani iz baze podatkov. Dnevniki se ciklično polnijo – ko je prvi napolnjen, se izvede preskok na drugega, ko je zadnji

poln, se izvede preskok na prvega. Osnovna naloga dnevnikov je preprečiti izgubo podatkov zaradi nedokončanega dela, ki bi ga moral izvesti proces za shranjevanje DBWn. Izpad se lahko zgodi zaradi več vzrokov, med drugim zaradi napake na računalniku, zaradi izpada električne energije ipd. Dnevnike za obnovo podatkov ažurira proces LGWR. Ažuriranje se opravi:

- po potrditvi vsake transakcije,
- zaradi povečanja dnevnika nad 1/3 velikosti,
- zaradi preseženih 1 MB spremenjenih podatkov v spominu baze,
- pred pisanjem procesa DBWn.

Vsi našteti kriteriji kažejo na to, da so dnevniki zelo obremenjena skupina datotek. Priporočljivo je, da se nahajajo na hitrih in od podatkov ločenih diskih. Zaradi njihove pomembnosti je tako kot pri kontrolni datoteki priporočljivo, da obstaja podvojena skupina dnevnikov, ki se nahaja na svojih diskih.

Arhivski dnevniki za obnovo podatkov

Ena izmed omenjenih lastnosti aktivnega podatkovnega skladišča je zagotavljanje njegove razpoložljivosti, kar pomeni delovanje brez prekinitev. Varnostno kopiranje podatkov se mora izvajati med delovanjem baze. Oracle omogoča to z uporabo tako imenovanega delovanja v načinu ARCHIVELOG. V prejšnjem odstavku opisani dnevniki se, ko so polni, prepišejo na rezervno lokacijo. Tako nastanejo arhivske kopije dnevnikov, ki pridejo v poštev pri restavriranju varnostnih kopij. Priporočljivo je, da se arhivske kopije shranjujejo na rezervni medij – npr. na trak.

Parameterska datoteka

Vsebuje podatke, ki so pomembni za delovanje baze. Pri zagonu instance se najprej vedno preveri vsebina te datoteke. Iz nje se dobi podatke o dodeljenem pomnilniku, lokaciji kontrolne datoteke, imenu baze in instance ipd. Med delovanjem se morebitne spremembe parametrov baze shranjujejo v to datoteko, s čimer se zagotovi, da se bo baza ob morebitnem ponovnem zagonu obnašala isto, kot se je pred ustavitvijo.

Arhivske kopije datotek

Zadnjo skupino datotek predstavljajo datoteke, ki so potrebne za morebitno rekonstrukcijo baze v primeru napake. Skupino sestavljajo parametrska, kontrolna in podatkovne datoteke, ki so bile ustvarjene ob trenutku arhiviranja sistema.

4.2.2.2 Logična podatkovna struktura baze

Logična podatkovna struktura je neodvisna od sprememb v fizični podatkovni strukturi. Uporabniki baze dostopajo do logične podatkovne strukture, ki jo sestavljajo:

- tablespace objekti,
- podatkovni bloki,
- extenti,
- segmenti.

Tablespace objekti

Vsako baza je sestavljena iz ene ali več logičnih enot, ki se imenujejo tablespace. Tablespace predstavlja skupino, ki logično združuje medsebojno podobne podatke. Pri generiranju baze se vedno generirata sistemska tablespace objekta, imenovana SYSTEM in SYSAUX. Naloga sistemskih tablespaceov je shranjevanje metapodatkov, ki jih baza potrebuje za svoje delo. Ti metapodatki se v Oraclu imenujejo podatkovni slovar (angl. *data dictionary*). Uporabniški podatki se shranjujejo v uporabniške tablespace, ki so običajno precej večji. Vsak tablespace sestavlja ena ali več podatkovnih datotek na disku. Pomembna lastnost tablespace objektov je ta, da so lahko dosegljivi ali nedosegljivi za uporabnike. Na ta način se lahko odpira ali zapira vsebinsko logične enote, kar olajšuje izvajanje skrbniških operacij na sistemu.

Podatkovni bloki

En podatkovni blok je najmanjša enota shranjevanja podatkov. Velikost bloka določa parameter DB_BLOCK_SIZE. Glede na poslovne potrebe se lahko velikosti blokov razlikujejo. Oracle omogoča pet različnih velikosti blokov.

Extenti

Predstavljajo naslednji višji logični nivo. Naloga extentov je logično združevanje zaporednih podatkovnih blokov. Oracle dodeljuje dodaten prostor na disku v velikosti extenta. Ker se extenti dodeljujejo po potrebi, niso nujno zaporedni.

Segmenti

Extenti so združeni v segmente. Segment je prostor, ki je dodeljen določeni podatkovni strukturi na bazi (tabeli, indeksu ...). Oracle pozna štiri vrste segmentov:

- podatkovne segmente za shranjevanje podatkov,
- indeksne segmente za shranjevanje indeksov,

- segmente rollback za pomoč pri delu s transakcijami,
- začasne segmente za potrebe sortiranja podatkov med izvajanjem stavkov SQL.

4.2.2.3 Pomnilniška struktura

Oracle uporablja pomnilnik za shranjevanje različnih podatkov, ki so potrebni za nemoteno delovanje strežnika. Pomnilniška struktura se deli na dva dela:

- Pomnilnik sistema (angl. *System Global Area, SGA*)
- Pomnilnik posameznih procesov (angl. *Program Global Area, PGA*)

SGA

Sistemski pomnilnik je prostor v pomnilniku računalnika, ki vsebuje podatke in kontrolne informacije. Oracle dodeli SGA prostor ob zagonu strežnika in ga sprosti ob ugasnitvi le-tega. Pomnilnik SGA je skupen vsem strežniškim in uporabniškim procesom. V primeru instalacije RAC ima vsak strežnik Oracle svoj SGA. Pomnilnik SGA je razdeljen na več podstruktur, od katerih so najpomembnejše:

- **Vmesni pomnilnik podatkovne baze**. Vsebuje najmlajše podatkovne bloke, ki jih je proces DBWn prebral iz podatkovnih datotek. Vsebovani so tako spremenjeni kot tudi nespremenjeni bloki. Večji ko je ta prostor, manj pogost je dostop do podatkovnih datotek – to pa vpliva na povečanje hitrosti.
- **Vmesni pomnilnik dnevnikov za obnovo podatkov**. Shranjuje vse spremembe, ki so se zgodile nad podatki. Ko so spremembe potrjene, jih proces LGWR zapiše v dnevnike za obnovo podatkov.
- **Skupni pomnilnik**. Vsebuje strukture, ki si jih delijo uporabniki. Ena izmed njih so stavki SQL, ki so bili uporabljeni.

PGA

Pomnilnik PGA pripada posameznim procesom. Prostor se dodeli, ko se proces zažene, in sprosti, ko se proces zaključi. Pomnilnik vsebuje podatke, ki jih potrebuje proces za izvedbo neke aktivnosti, npr. vrednosti programskih spremenljivk.

4.2.2.4 Procesi

Procese, ki jih srečamo v okolju Oracle, delimo na dve skupini:

- uporabniške procese (sprožajo jih uporabniki),
- strežniške procese (delujejo avtonomno in neodvisno od ljudi).

Uporabniški procesi

Odvisno od konfiguracije lahko uporabniki uporabljajo individualne (angl. *dedicated*) ali skupne (angl. *shared*) uporabniške procese. Individualni uporabniški procesi so namenjeni zahtevnejšim uporabnikom, ki potrebujejo več virov na strežniku. Skupni uporabniški procesi so primerni za običajne uporabnike z operacijami, ki povprečno obremenjujejo sistem. Če se to preslika na področje podatkovnega skladiščenja, se individualni procesi uporabljajo za strateške, skupni pa za operativne poizvedbe.

Strežniški procesi

Predstavljajo drugo skupino procesov. Odvisno od konfiguracije sistema je na voljo različno število procesov, najpomembnejši izmed njih pa so:

- SMON ali strežniški proces, ki skrbi za pravilen zagon baze.
- PMON ali nadzornik procesov, ki skrbi za sproščanje virov, ki so jih zasedli prekinjeni uporabniški procesi.
- Procesi DBWn skrbijo za shranjevanje spremenjenih podatkov iz začasnega spomina v podatkovne datoteke. Ker SUBP Oracle spremenjene podatke shranjuje v dnevnike za obnovo podatkov, se procesi DBWn izvajajo samo takrat, ko se pojavi zahteva po dodatnemu prostoru v spominu SGA. Takrat procesi z uporabo algoritma LRU (Last Recently Used) zapišejo v podatkovne datoteke najmanj pogosto uporabljene podatkovne bloke.
- Proces LGWR skrbi za sinhronizacijo dnevnikov za obnovo podatkov s stanjem v spominu.
- Procesi ARCn skrbijo za kopiranje arhivskih dnevnikov.
- Proces CKPT skrbi za sinhronizacijo baze. V kontrolne datoteke in v glave podatkovnih datotek shrani sinhronizacijski podatek SCN, ki v primeru izpada sistema služi za njegovo ponovno postavitev v stanje zadnje shranjene transakcije.
- Proces RECO skrbi za reševanje napak pri distribuiranih transakcijah.

4.3 ANALIZA PRIPRAVLJENOSTI ZA PODPORO AKTIVNEMU PODATKOVNEMU SKLADIŠČENJU

V tem poglavju bom na prototipu prikazal primer implementacije aktivnega podatkovnega skladišča, podprtega s funkcionalnostmi SUBP Oracle 10g. Prototip podatkovnega skladišča pokriva proces internetne prodaje. Internetna prodaja je

nova tržna pot, ki jo do množične uvedbe interneta nismo poznali. Zanjo je značilno, da lahko uporabnik, ne glede na to, kje se nahaja, s pomočjo računalnika, priključenega na internet, nakupuje izdelke in storitve. Te zlaga v virtualno nakupovalno košarico, ki jo na koncu postopka plača s kreditno kartico. Podjetje, ki uporablja internetno trgovino, kupcu kupljene izdelke/storitve pošlje po pošti.

4.3.1 Opis poslovnega procesa pred implementacijo

Tako kot običajne trgovine se tudi internetne trgovine srečujejo s čedalje večjo konkurenco. Trgovci poizkušajo kupce v svojih trgovinah obdržati s tem, da vsakega kupca obravnavajo individualno. V klasični trgovini se je že dobro uveljavil koncept kartic zvestobe. Kupci pri nakupu priložijo omenjeno kartico, ki služi kot identifikator. Podjetje na ta način pridobi informacijo o kupcu in prodane izdelke namesto na neznanega opredeli na točno določenega kupca. Nad prodajnimi podatki se v določenih intervalih izvedejo računalniške obdelave. Te kupcem, ki so z nakupi dosegli določene zneskovne razpone, dodelijo popust v obliki dobroimetja, ki ga lahko kupec porabi pri nakupih v naslednjem obdobju.

Rešitev je na prvi pogled korektna, vendar pa lahko v sebi združuje več slabosti, ki so kupcem na prvi pogled nevidne. Kot primer opisujem velikega slovenskega trgovca, ki ga bom namenoma pustil neimenovanega. Pri njem se vsak nakup pretvori v točke. Točka je vredna določen znesek. Primer pri vrednosti točke 4 €:

- znesek nakupa znaša 3,9 €, kupec prejme 0 točk,
- znesek nakupa znaša 15,9 €, kupec prejme 3 točke.

Tako pridobljene točke se seštevajo določeno obdobje. Po preteku tega obdobja se vsakemu seštevku, odvisno od tega, v katerem razponu se nahaja, dodeli dobroimetje. Tega kupec koristi v naslednjem obdobju. Stanje točk na kartici zvestobe pa se postavi na 0.

Po mojem mnenju gre pri opisanem primeru za agresivno obliko modela, saj kupce na več nivojih dobesedno sili k nakupom, in sicer:

- Kupci se morajo pri vsakem nakupu truditi, da presežejo znesek, ki jim da novo točko.
- Pri kupcih je proti koncu obdobja prisotna nakupovalna »mrzlica«, ki je posledica želje po doseganju določenega števila točk, ki bo prineslo dobroimetje.
- Ne glede na to pa so vsi kupci na koncu obdobja »nagrajeni« z izbrisom stanja na kartici zvestobe.

Trgovec lahko takšen poslovni model opravičuje z zahtevnostjo obdelav. Te se ne izvršujejo v realnem času, ampak paketno. Po mojem mnenju bi lahko opisani

model izboljšal z uvedbo aktivnega poslovnega skladišča, ki bi dinamično izračunaval popuste.

4.3.2 Opis spremenjenega poslovnega procesa

V nadaljevanju bom opisal prototip, ki s pomočjo aktivnega podatkovnega skladiščenja rešuje slabosti opisanega modela. Pokriva proces internetne prodaje, ki ima to posebnost, da pri vsaki izstavitvi računa dinamično dodeljuje popust, katerega višina je odvisna od višine nakupov v preteklih X letih. Gre za načelo operativne poslovne inteligence – uporabo povratne zanke.

Opisana rešitev ima v primerjavi z v prejšnjem poglavju omenjeno rešitvijo vsaj naslednje tri prednosti:

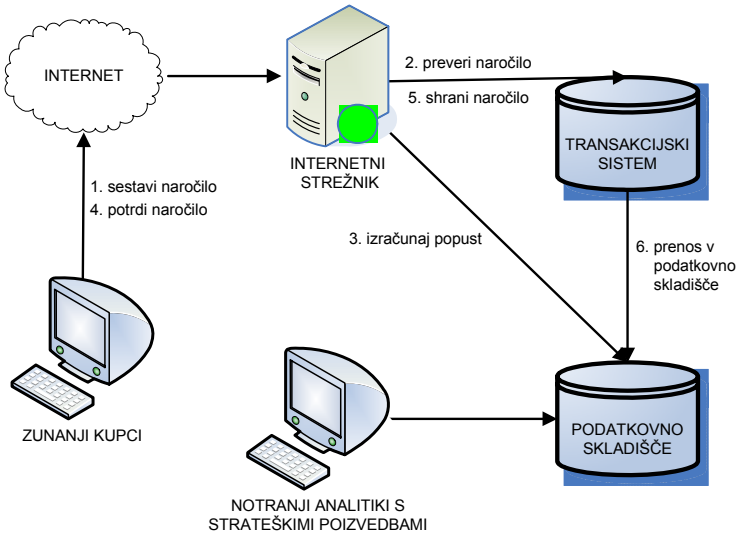
- Seštevajo se zneski, ne točke. Tako ne prihaja do neupoštevanja določenih delov nakupa.
- Obdobje izračunavanja je daljše in drseče. Ne prihaja do inicializacije stanja po preteku obračunskega obdobja.
- Kupci popuste občutijo takoj, pri vsakem nakupu.

Sam postopek nakupovanja je podoben običajnim nakupom v internetni trgovini (slika 9). Kupec preko interneta dostopa do prodajne strani podjetja. Tam se identificira s pomočjo uporabniškega imena in gesla. Sledi postopek nakupovanja, kjer kupec v virtualno košarico dodaja izdelke, ki jih želi kupiti. Po izboru želenih izdelkov sledi zaključek nakupovanja. V tem koraku se vključi aktivno podatkovno skladišče. Program s pomočjo poizvedbe nad aktivnim podatkovnim skladiščem ugotovi znesek preteklih nakupov v preteklih X letih. Na podlagi dobljenega zneska mu določi popust, ki se doda v trenutno nakupovalno košarico. Sledi vpis podatkov o načinu plačila in njihov prenos preko interneta s pomočjo varne strani. Nakup je tako končan in kupec samo še čaka na izdelke, ki jih kmalu prejme po pošti.

Po opravljenem nakupu se podatki prenesejo v podatkovno skladišče, kjer so na voljo internim analitikom in morebitnim novim nakupom.

Po mojem mnenju gre pri opisanem modelu za kupcem bolj prijazen model nakupovanja, ki ga bodo verjetno morala v podobnih oblikah, zaradi čedalje večjega zavedanja kupcev, vpeljevati tista podjetja, ki bodo želela in delala na partnerskem odnosu s kupci.

Slika 9: Shematski prikaz vloge podatkovnega skladišča v proučevanem prototipu



Vir: Avtorjevo lastno delo

4.3.2.1 Tehnične zahteve

Prototip je poenostavljen do te mere, da na eni strani omogoča preprosto razumevanje problematike, obenem pa dovoljuje, da ga je možno z majhnimi modifikacijami, ki so značilne za posamezno branžo, implementirati tudi v produkcijskih okoljih. Zahteve sistema so bile:

- Enotno centralno podatkovno skladišče.
- Podatki morajo biti na voljo 24/7, zagotovljena mora biti ponovna vzpostavitev sistema v primeru katastrofe.
- Podatek o naročilu mora biti analizam na voljo največ 10 minut po nastanku tega dogodka v transakcijskem sistemu. Razlog za dovoljen 10-minutni zaostanek se nahaja v predpostavki, da kupec običajno ne opravi dveh zaporednih nakupov. Če pa se že odloči za zaporedno nakupovanje, pa novega nakupa ne opravi vse do tedaj, dokler za prejšnjega ne prejme potrditve po elektronski pošti. Ta pa lahko do naslovnika potuje od nekaj sekund do nekaj minut.
- Podatkovno skladišče mora podpirati operativne in strateške poizvedbe, ki prihajajo iz različnih aplikacij in paketnih obdelav. Pri tem morajo imeti operativne poizvedbe prioriteto pred strateškimi poizvedbami.
- Sistem mora uspešno obvladovati veliko število transakcij. Trenutna količina je 100.000 postavk dnevno, predvidena rast podatkov približno 20 % letno. Predpostavka, ki je na videz močno predimenzionirana, se lahko hitro izkaže kot pravilna. Ker internet ne pozna meja, se lahko zgodi, da

trgovina, četudi je bila zasnovana za lokalno geografsko območje, preraste v globalno internetno trgovino.

Proces internetne prodaje analiziranega prototipa so sestavljali trije tehnični sklopi:

- Internetni strežnik, na katerem se izvaja internetna aplikacije za sprejem naročil.
- Transakcijski sistem, ki je beležil podatke o naročilih. Transakcijski sistem uporablja SUBP Oracle 10g.
- Podatkovno skladišče, katerega naloga je bila shranjevanje podatkov in omogočanje dostopa do njih za potrebe strateških ter operativnih poizvedb.

4.3.2.2 Pristop izvedbe podatkovnega skladišča

V svetu podatkovnega skladiščenja srečamo tri načine implementacije:

- Relacijsko (ROLAP)
- Multidimenzionalno (MOLAP)
- Hibridno (HOLAP)

ROLAP

Uporablja se ga, kadar imamo opravka z velikimi količinami podatkov. Pri tem pristopu so podatki shranjeni v relacijski podatkovni bazi, do njih pa se dostopa s poizvedovalnim jezikom SQL. Slabosti ROLAP-a sta predvsem njegova počasnost pri izvajanju poizvedb in omejenost nabora analitičnih funkcij. Novejši sistemi za upravljanje baz podatkov te pomanjkljivosti rešujejo z agregatnimi tabelami in nadgradnjami jezika SQL.

MOLAP

Implementacija MOLAP se običajno uporablja pri majhnih količinah podatkov. Podatki so tu namesto v relacijskih podatkovnih bazah shranjeni v večdimenzionalnih strukturah – kockah. Kocke vsebujejo predagregirane podatke, zato so poizvedbe nad njimi hitre. Ker je osveževanje kock dolgotrajen postopek, njihova uporaba ni primerna za velike količine podatkov.

HOLAP

Prednosti pristopov ROLAP in MOLAP združuje pristop HOLAP. Atomarni podatki so pri tem pristopu shranjeni v relacijski bazi, agregirani podatki pa v večdimenzionalnih strukturah kock MOLAP.

Ker so zahteve sistema predvidevale delo z velikimi količinami podatkov, sem se odločil za izvedbo podatkovnega skladišča ROLAP. Postopke okoli izvedbe sem

glede na vrsto dela razdelil na postavitev infrastrukture in postavitev podatkovnega skladišča.

4.3.3 Postavitev infrastrukture

Zahteva po neprekinjenem delovanju (24/7) in možnost istočasnega izvajanja operativnih ter strateških poizvedb je narekovala razmišljanje o sistemu, ki bi uporabljal paralelizem. Oracle paralelizem na nivoju strojne opreme zagotavlja z uporabo gruče strežnikov RAC.

4.3.3.1 Oracle RAC

RAC je oblika konfiguracije sistema, kjer več strežnikov dostopa do skupne baze podatkov. Gre za Oraclovo izvedbo paralelizma s pomočjo gruče večprocesorskih strežnikov, ki sem ga opisal v poglavju 2.5.1.3. Konfiguracija RAC je predstavnik tako imenovane »deli vse« arhitekture, kjer vsi strežniki v gruči dostopajo do skupne baze podatkov. Konfiguracija podpira baze podatkov, ki temeljijo na:

- samostojnem diskovnem polju (angl. *Storage Area Network, SAN*),
- omrežnem priključenem diskovju (angl. *Network Attached Storage, NAS*).

NAS in SAN

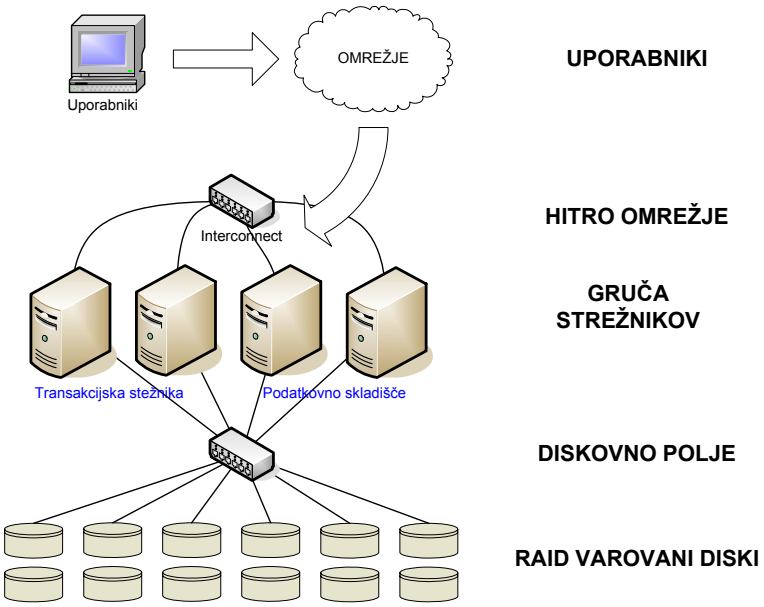
NAS oziroma omrežno priključeno diskovno polje je praktično samostojen računalnik, katerega naloga je skladiščenje podatkov za potrebe drugih računalnikov v omrežju. Slednji ga lahko vidijo na različne načine − kot FTP-strežnik, datotečni strežnik ali računalnik znotraj lokalnega omrežja. NAS je zaradi nižje cene in enostavnejšega vzdrževanja primeren za manj zahtevne implementacije.

SAN predstavlja samostojen sistem za skladiščenje podatkov, ki je priklopljen neposredno na strežnik oziroma računalnik. Če hočejo drugi uporabniki dostopati do podatkov, shranjenih na njem, gre celotna komunikacija prek tega računalnika. SAN praviloma zahteva hitre povezave med diskovnim poljem in strežnikom, te so v zadnjem času optične, malo starejše pa tipa SCSI. SAN predstavlja najnaprednejšo obliko shranjevana podatkov in je primeren za najzahtevnejše uporabnike.

Slika 10 prikazuje arhitekturo prototipa, temelječega na konfiguraciji RAC Oracle baze. Na sliki je vidno dodatno hitro omrežje. Naloga tega omrežja je odpravljanje problemov sinhronizacije spremenjenih blokov, ki sem jih omenil na strani 38 v poglavju 3.4.1.3. To hitro omrežje zagotavlja, da so vsi vmesni pomnilniki

strežnikov v gruči vidni kot en skupen vmesni pomnilnik (Barb, 2006; Freeman 2004).

Slika 10: Arhitektura Oracle RAC



Vir Prirejeno po Barb, 2005

Implementacija na prototipu

Oracle omogoča, da se posameznim strežnikom dodeljujejo posamezna opravila glede na tip obremenitev. Prva dva strežnika sem namenil transakcijskim obdelavam, druga dva pa potrebam podatkovnega skladišča. Strežnika podatkovnega skladišča sem ločil glede na tip poizvedb, ki jih bosta izvajala. Prvemu je bilo dodeljeno izvajanje občasnih, vendar zahtevnih strateških poizvedb in izvajanje aplikacije za polnjenje podatkovnega skladišča. Drugi strežnik je bil namenjen izvajanju enostavnih operativnih poizvedb, ki bi jih neprenehoma izvajala internetna aplikacija za prejemanje naročil.

Instalacija RAC podpira implementacijo rezervnih strežnikov za primer izpada (angl. *fail-over*). V praksi to izgleda tako, da delo primarnega strežnika v primeru izpada prevzame sekundarni strežnik, ki je bil predhodno določen. Sprememba se zgodi avtomatsko brez posredovanja administratorja sistema. Vsakemu izmed vseh štirih strežnikov sem dodelil sekundarni strežnik, ki je bil namenjen servisiranju poizvedb v primeru odpovedi primarnega strežnika. Varovanje je dodeljeno v parih 12, 21, 34 in 43, pri čemer prva številka pomeni zaporedno številko primarnega in druga zaporedno številko sekundarnega strežnika.

Z zgoraj opisano konfiguracijo sistema se je pridobilo naslednje:

- Povečala se je razpoložljivost sistema tako pri načrtovanih kot tudi pri nenačrtovanih izpadih sistema.
- Zagotovilo se je primerno dodeljevanje virov posameznim poizvedbam. Z deljenjem na različne strežnike se je preprečilo, da bi strateška poizvedba uporabila vse vire in bi zaradi čakanja na rezultat operativne poizvedbe bilo zaustavljeno delo z internetno aplikacijo.
- Zagotovil se je paralelizem.
- Povečala se je razširljivost sistema, saj bi se ob povečani rasti zahtev v gručo dodal dodatni strežnik, ki bi odpravil ozko grlo. Takšen način dodajanja pa zmanjšuje stroške lastništva sistema (angl. *Total Cost of Ownership, TCO*), saj nadgradnje niso povezane z menjavo obstoječih sistemov.

Opisana konfiguracija, ki je bila sposobna odpravljati večino izpadov sistemov, pa ni bila zmožna zagotoviti delovanja sistema v primeru katastrofe. Za katastrofo, naj bo to človeškega ali naravnega izvora, je značilno fizično uničenje podatkovnega centra. Obstaja več pristopov, kako zagotoviti, da bo sistem varen pred izpadi. Oracle ponuja rešitev v obliki produkta Data Guard.

4.3.3.2 Data Guard

Data Guard je upravljalna, nadzorna, avtomatizacijska infrastruktura, katere naloga je izdelava, vzdrževanje in nadzor rezervnih SUBP z namenom, da se primarni SUBP obvaruje pred izpadi, napakami in katastrofami. Varovanje se zagotavlja s kopiranjem podatkov iz produkcijskega okolja na drugo lokacijo, ki se lahko nahaja kjerkoli na svetu. Edini pogoj je, da sta oba SUBP medsebojno vidna preko internetnih povezav. V primeru izpada produkcijske strani Data Guard zamenja vlogi baz tako, da rezervna stran postane produkcijska in omogoči uporabnikom sistema nemoteno nadaljevanje dela (Shupmann, 2006).

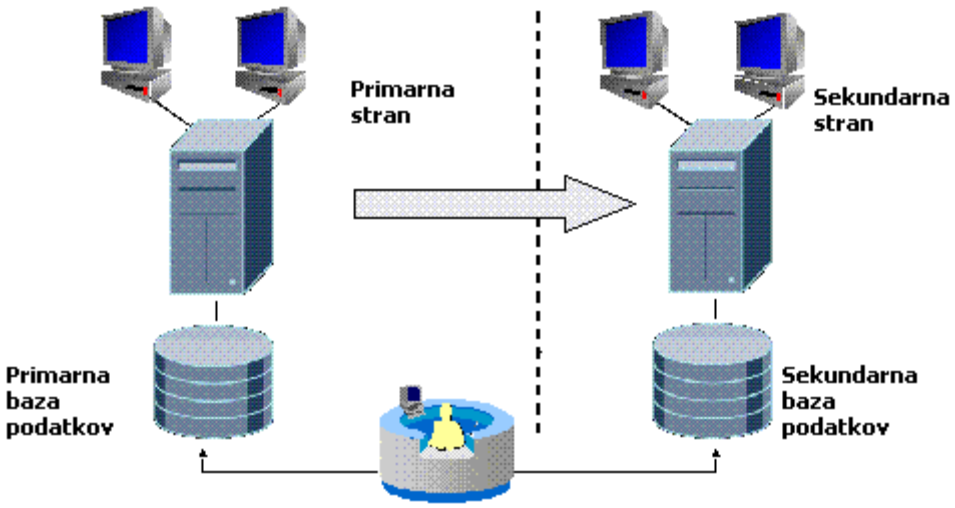
Data Guard podpira konfiguracijo do devetih rezervnih sistemov. Če želimo zagotoviti čim večjo razpoložljivost sistema, je priporočljivo, da se rezervni sistem nahaja na geografsko oddaljeni lokaciji, najbolje na drugi tektonski plošči, s čimer se izloči vpliv potresov kot geografsko najobsežnejše oblike katastrofe.

Slika 11 prikazuje glavne komponente sistema. Strežnika na sliki sta lahko zamenjana s konfiguracijo sistema RAC, kar sem v nadaljevanju uporabil pri analizi preučevanega prototipa.

Običajno je, da se na rezervni strani podatkovna baza zgradi iz kopije podatkovne baze na primarni strani. Podatki se v rezervno bazo dodajajo na dva načina:

- s pomočjo dnevnikov za obnovo podatkov, pri čemer gre za postopek, ki je podoben rekonstrukciji baze podatkov – procesa povrnitve podatkov v čas pred napako sistema (angl. *recovery*). Na ta način se zagotovi fizična kopija primarne baze podatkov,
- s stavki SQL, pridobljenimi iz dnevnikov za obnovo podatkov, s čimer se zagotovi logična kopija primarne baze podatkov.

Slika 11: Shematični način delovanja infrastrukture Data Guard



Vir:prirejeno po Schupman, 2006

Data Guard podpira dva načina menjav primarne in sekundarne strani. Prvi se imenuje preklop (angl. *switch-over*) in se ga uporablja v primeru načrtovanih izpadov (npr. zaradi vzdrževalnih del) primarnega sistema. Po izvedbi vzdrževalne operacije na primarni strani se ponovno zgodi preklop in konfiguracija se povrne v prvotno stanje. Drugi način menjav pa se zgodi zaradi katastrofe na primarni strani (angl. *fail-over*). Po avtomatskem preklopu na sekundarno stran, vrnitev nazaj ni več možna. Zato je po odpravi katastrofe na primarni strani potrebno ponovno nastaviti infrastrukturo Data Guard. Data Guard omogoča tri nivoje zaščite primarnih SUBP:

- Z **nivojem maksimalne zaščite** podatkov se podatki sinhrono prenašajo s primarne na sekundarno stran. Transakcija je v primarni bazi podatkov potrjena šele takrat, ko je predhodno potrjena v vseh (v enem ali več) sekundarnih bazah podatkov.
- Pri **nivoju maksimalne razpoložljivosti** sistema se od nivoja maksimalne zaščite loči po tem, da tudi v primeru izpada enega izmed sekundarnih SUBP primarna baza nadaljuje z delom. Po vzpostavitvi sekundarne strani v delujoče stanje se vsebina avtomatsko sinhronizira s primarno stranjo.

- **Nivo maksimalne zmogljivosti** ne pogojuje obstoja sekundarne strani. Primarna stran deluje neodvisno od tega, ali je sekundarna stran delujoča ali ne. Sekundarna stran zajema podatke iz arhivskih dnevnikov za obnovo podatkov na asinhroni način.

Implementacija na prototipu

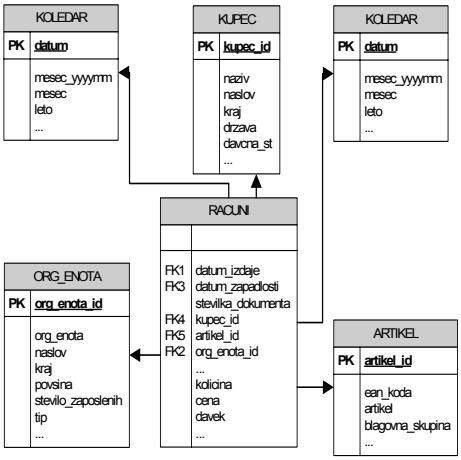
Prototip je podpiral mehanizem Data Guard. Na primarni strani ga je sestavljala konfiguracija RAC štirih strežnikov (opisano v prejšnjem poglavju), na sekundarni strani pa dva strežnika, prav tako v konfiguraciji RAC. Sekundarna stran je bila zmogljivostno okrnjena, predvidevala je, da prvi sekundarni strežnik beleži transakcijske dogodke in izvaja operativne poizvedbe nad podatkovnim skladiščem. Razlog za združitev obeh procesov pod okrilje enega strežnika izhaja iz podobnosti poizvedb. Drugi sekundarni strežnik je bil namenjen za potrebe polnjenja podatkovnega skladišča in izvajanje strateških poizvedb. Lokacija sekundarnega strežnika bi se morala v primeru produkcijske implementacije okolja nahajati na oddaljeni lokaciji. Konfiguracija Data Guard je bila nastavljena na nivo zaščite z maksimalno razpoložljivostjo sistema.

4.3.4 Postavitev podatkovnega skladišča

4.3.4.1 Logični podatkovni model

Logični podatkovni model sestavlja pet dimenzijskih tabel in ena tabela dejstev (slika 12). Ker so podatki v dimenzijskih tabelah denormalizirani, lahko govorimo o zvezdastem modelu (angl. star schema) (Scalzo, 2003).

Slika 12: Logični podatkovni model prototipa



Vir: avtorjevo lastno delo

Dimenzije

Pri oblikovanju dimenzij sem pozornost usmeril k čim večji optimizaciji prostora na disku. Vsakemu tekstovnemu polju sem priredil tip VARCHAR2, primarnim ključem pa podatkovni tip DATE ali NUMBER. Uporaba datumskih ali numeričnih podatkovnih tipov za potrebe ključev je pomembna, ker se na ta način zmanjša potreben prostor za shranjevanje podatkov v tabeli dejstev. Primer izpeljanega ključa je na sliki 13 viden v dimenziji ARTIKEL, kjer bi kot ključ zadoščalo tudi polje EAN_KODA, vendar bi podatki zaradi tekstovnega podatkovnega tipa na diskih zasedali več prostora. Ena izmed osnovnih značilnosti dimenzijskih tabel je tudi ta, da se podatki v njih lahko spreminjajo skozi čas. Glede na to, kako se ohranja zgodovina, se je v svetu uveljavil pristop počasi spreminjajočih dimenzij (Kimball, 2002). Ker te ne vplivajo na osnovni namen testiranja prototipa, sem omenjeno metodologijo namenoma izpustil iz modela (Chyran, 2005; Hobbs, 2004).

Tabela dejstev

Tabela dejstev vsebuje atomarne podatke o vsaki opravljeni transakciji. V transakcijskem sistemu je običajno, da so podatki o računih zaradi normalizacije razdeljeni v dveh objektih, npr. RACUNI_GLAVA in RACUNI_POSTAVKE, ki sta medsebojno običajno povezana s poljem STEVILKA_DOKUMENTA, ki je v obeh tabelah indeksirano. Običajno se do podatkov dostopa posamezno (vsak račun posebej). Zaradi tega je dostop hiter. Nad podatkovnim skladiščem se nasprotno izvajajo poizvedbe, ki zahtevajo agregirane podatke. Te se morajo izvesti čim hitreje. Pristop iz transakcijskih sistemov se tu ne obnese, ker pri poizvedbi tega tipa prihaja do velikega števila povezovanj med tabelami, kar pa vpliva na daljši čas izvajanja. Zmanjševanje števila povezav med tabelami predstavlja glavni razlog za uporabo denormalizacije v okolju podatkovnega skladiščenja. Denormalizacija je prisotna tako v dimenzijah kot tudi v tabeli dejstev.

V bistvu gre za kompromis med časom izvajanja poizvedb in zasedenostjo prostora na diskih. Transakcijski sistemi, ki so običajno normalizirani v tretji normalni formi, zagotavljajo minimalno redundanco podatkov in preprečujejo anomalije pri ažuriranju podatkov. Vse to v zameno za daljši čas izvajanja poizvedb v primerjavi s podatkovnim skladiščem, ki pa hitrejše poizvedbe plačuje s povečano porabo diskovnega prostora.

Tabela dejstev je zaradi svoje značilnosti beleženja zgodovinskih poslovnih dogodkov največji porabnik diskovnega prostora. Obstaja več načinov zmanjševanja porabe diskovnega prostora. Na logičnem nivoju (neodvisno od ponudnika SUBP) je potrebno slediti dejstvu, da lahko vsak prihranjen byte pri definiciji posameznega polja v tabeli dejstev pomeni finančni prihranek zaradi manjših diskovnih potreb celotnega sistema. Optimizacija vrača izredne finančne prihranke v področjih, kjer so tabele dejstev zaradi narave posla zelo velike. Eno izmed takšnih področij predstavljajo telekomunikacije, kjer se tabele dejstev

dnevno povečujejo za milijone zapisov s podatki o opravljenih telefonskih pogovorih. Znan je primer iz prakse, kjer je prenova tabele dejstev v podjetju AT&T leta 1998 prihranila milijone $ (Sybase Techwave, 1999).

V logičnem podatkovnem modelu sem strmel k temu, da bi se v tabeli dejstev v celoti izognil tekstovnim podatkovnim tipom. Za vsa polja, tako reference na dimenzije kot tudi mere, sem zato izbral datumski ali numerični podatkovni tip.

4.3.4.2 Fizični podatkovni model

Fizični podatkovni model predstavlja preslikavo logičnega podatkovnega modela v konkretno bazo podatkov. Značilnost aktivnega podatkovnega skladišča – omogočanje istočasnega izvajanja strateških in operativnih poizvedb – je narekovala uporabo posebnih pristopov, določene izmed njih sem omenil v poglavju 4.2.2. Preden nadaljujem s fizičnim podatkovnim modelom, bom opisal še nekaj funkcionalnosti, ki sem jih potreboval pri implementaciji le-tega.

Particioniranje objektov

Particioniranje objektov je osnovna funkcionalnost, ki jo Oracle ponuja pri podatkovnem skladiščenju. Particioniranje pomeni deljenje objektov (tabel, indeksov ali materializiranih pogledov) v manjše, bolj obvladujoče enote, ki se navzven obnašajo kot celota. Prednosti particioniranja se kažejo na dveh področjih (Lane, 2002):

- Administratorska opravila, kot so polnjenje podatkov, izgradnja indeksov, osveževanje statistik, brisanje podatkov s particioniranjem, pridobijo na obvladljivosti.
- Uporabniške poizvedbe se lahko pohitrijo, saj se v primeru pogojev v stavkih SQL poizvedbe izvajajo samo na particijah, ki vsebujejo vrednosti, navedene v pogoju. Ta postopek se imenuje dinamično omejevane particij (angl. *Dynamic Partition Pruning*).

Običajna praksa je, da se particionirajo tabele dejstev. Particioniranje se izvaja nad določenim stolpcem v tabeli, npr. poljem DATUM_IZDAJE v našem primeru. Glede na način particioniranja Oracle 10g ponuja več metod:

- Particioniranje v razponih (angl. *range partitioning*). Običajno je, da se ga uporablja pri časovnih obdobjih, kot so leta ali meseci, in posledično zato velja za najpogostejšo obliko particioniranja.
- Particioniranje po hash algoritmu (angl. *hash partitioning*). Uporablja se ga, če želimo enakomerno distribucijo podatkov.

- Particioniranje z navedbo diskretnih vrednosti za vsako particijo (angl. *list partitioning*). Uporablja se ga v primeru, če se določene vrednosti polja, po katerem je particionirana tabela, pojavljajo večkrat.
- Kombinirano particioniranje naštetih metod (angl. *composite partitioning*). Predstavlja dvonivojsko obliko particioniranja. Na vsakem izmed nivojev se uporablja ena izmed prej naštetih metod.

Zgoščevanje podatkov

Čeprav z omenjenim particioniranjem olajšamo delo pri velikih količinah podatkov, z njim ne moremo zmanjšati stroškov, povezanih s shranjevanjem. Velik del stroškov, ki so povezani s shranjevanjem, je neposredno povezan s cenami diskovnega prostora (Graefe, 1991). Oracle 10g ponuja zgoščevanje podatkov in se od običajnih algoritmov zgoščevanja podatkov razlikuje v tem, da je optimiziran strukturi relacijske baze. Zgoščevanje se odvija na nivoju posameznega bloka na disku tako, da se večkrat ponovljene vrednosti zamenjajo s kazalcem na seznam vrednosti, ki se nahaja na začetku vsakega zgoščenega bloka. Z zgoščevanjem podatku se pridobi dvoje:

- Zmanjša se poraba diskovnega prostora, kar neposredno vpliva na ceno podatkovnega skladiščenja.
- Poveča se hitrost izvajanja poizvedb. Vzrok za to je razmerje med časi branja in časi procesiranja zgoščenih blokov. Čeprav zgoščeni bloki potrebujejo več procesorskih moči, je to povečanje minimalno v primerjavi z velikim prihrankom časa pri dostopu do podatkov na disku.

Prihranek prostora je odvisen od strukture podatkov. Običajno se objekti zgostijo na od 1/2 do 1/3 začetne velikosti, v praksi pa so znane tudi zgostitve na 1/12 začetne velikosti (Ozbutun, 2005).

Indeksi

Indeksi so strukture, ki omogočajo hitrejši dostop do iskanih podatkov, saj se s pomočjo njih izognemo pregledu celotne tabele za posamezno poizvedbo. V transakcijskih sistemih se srečujemo z indeksi, ki so oblikovani v obliki B-drevesa (angl. *B-tree*), kjer so indeksi navadno mnogo manjši od tabele same. Ko se baza odloči za dostop do podatkov preko indeksov, so rezultati na voljo hitreje kot pri poizvedbi brez uporabe indeksov.

V podatkovnem skladišču se pojavi naslednja težava. Tabela dejstev je navadno nekaj tisočkrat večja od dimenzijskih tabel in od tabel v transakcijskih sistemih. Če bi na poljih postavili B-drevesne indekse, bi zaradi svoje zgradbe zasedli tudi do nekajkrat več prostora kot podatki sami. Zato se za indeksiranje polj uporabljajo binarni indeksi (angl. *bitmap*), ki imajo več prednosti:

- so manjši od običajnih indeksov,

- uporabljajo se lahko pri paralelnih poizvedbah,
- pospešijo večino ad-hoc poizvedb.

Na splošno velja, da je neko polje kandidat za postavitev binarnega indeksa, če predstavlja število različnih vrednosti v tem polju do 1 % vseh vrednosti v tabeli. To razmerje se imenuje kardinalnost (Hobbs, 2004; Rosenberg, 2006).

Materializirani pogledi

Materializirani pogledi se uporabljajo za agregiranje tabel, v katerih se nahajajo podatki, sešteti po najpomembnejših dimenzijah. Takšne tabele so običajno nekajkrat manjše od prvotnih tabel. Njihova uporabnost se kaže v tem, da predizračunavajo dolgotrajne in pogosto porabljene poizvedbe v samostojne objekte. Na ta način je možno pohitriti običajno dolgotrajne poizvedbe v podatkovnih skladiščih. Definicije materializiranih pogledov so podane v obliki stavkov SELECT SQL, ki lahko vsebujejo funkcije za agregiranje podatkov (npr. SUM, COUNT ...) in povezave med več tabelami. Na splošno obstajata dva načina osveževanja materializiranih pogledov:

- Avtomatsko osveževanje po vsaki potrditvi transakcije v tabeli dejstev (angl. *on commit*). Mehanizem zagotavlja vsebinsko enakost podatkov v tabeli dejstev in materializiranem pogledu. Slaba lastnost je obremenjevanje sistema zaradi stalnega predizračunavanja.
- Na zahtevo (angl. *on demand*). Materializirani pogledi se osvežijo v času, ko se sistem manj obremenjen, npr. ponoči. V zakup pa je potrebno vzeti razlikovanje vsebine med tabelo dejstev in materializiranim pogledom nad to tabelo.

Materializirani pogledi se lahko izračunavajo vsakič v celoti ali pa samo prirastek podatkov, kar je običajna praksa pri podatkovnem skladiščenju. Oracle ima v verziji 10g vgrajeno funkcionalnost, ki v primeru obstoja primernega materializiranega pogleda omogoča avtomatsko transformacijo stavka SQL v nov stavek (angl. *Query Rewrite*). Novi stavek se brez posredovanja uporabnika avtomatsko izvede nad materializiranim pogledom. Rezultat tega je hitrejša poizvedba (Hobbs, 2004).

Paralelno izvajanje poizvedb

Paralelno izvajanje predstavlja naslednji način povečevanja hitrosti v podatkovnem skladišču. Predpogoj za izvajanje paralelizma je ustrezna strojna infrastruktura, ki mora vsebovati vsaj dva procesorja. Izvajanje paralelizma v okviru podatkovnega skladiščenja pomeni prihranek časa pri naslednjih operacijah:

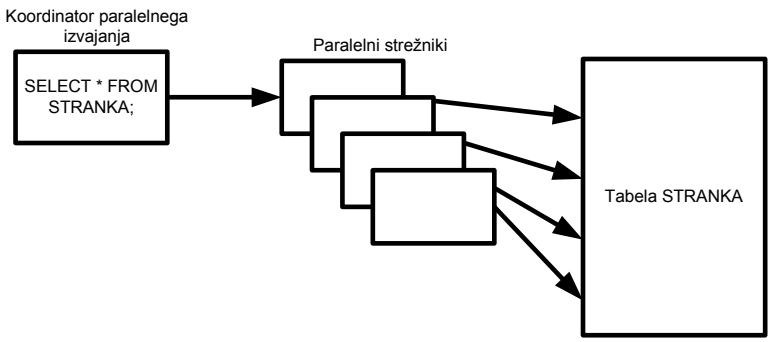
- poizvedbah po tabelah in indeksih,
- izgradnji in spreminjanju velikih indeksov,

- izgradnji in spreminjanju velikih tabel ali
- množičnih vnosih, spreminjanju in brisanju podatkov.

Ko uporabnik sproži izvajanje stavka SQL, se optimizator poizvedbe baze odloči, ali bo paralelizem povečal hitrost poizvedbe. V primeru pozitivne odločitve določi stopnjo paralelizma. Glavni koraki izvajanja paralelizma so naslednji (slika 13):

- osnovni proces postane koordinator za paralelno izvajanje,
- ta določi število paralelnih strežnikov, katerih število je omejeno z maksimalno stopnjo paralelizma,
- začne se izvajanje stavka – če je le mogoče, paralelno,
- ko se posamezni procesi zaključijo, rezultate združi osnovni koordinatorski proces in prikaže rezultat.

Slika 13: Delovanje paralelizma



Vir: prirejeno po Hobbs, 2004

Fizična implementcija

Prvi korak pri fizični implementaciji je bila postavitev ločenega tablespace prostora na disku, v katerem sem kasneje generiral vse objekte. Tablespaceu sem definiral velikost bloka 32k bytov. Razlog za izbiro največje velikosti bloka leži v dejstvu, da tipična poizvedba v podatkovnem skladišču prebere veliko več podatkov kot poizvedba v transakcijskem sistemu. Podatki se berejo v enotah, ki ustrezajo velikosti bloka. Preklop iz enega v drugi blok je potratna operacija (Oracle Metalink, 2006), zato je za poizvedbe nad podatkovnim skladiščem bolje, da se izvajajo v manjšem številu velikih kot v več številu manjših blokov (npr. manj blokov po 32 k, namesto več blokov po 2k bytov).

Izgradnja dimenzijskih tabel

Vsaki dimenzijski tabeli v logičnem modelu je bil dodeljen svoj segment v tablespace prostoru, namenjenem podatkovnemu skladiščenju. Za potrebe avtomatskega generiranja izpeljanih ključev sem na tabelah, ki to potrebujejo (ARTIKEL, KUPEC in ORG_ENOTA), izdelal sekvence. Sekvenca je samostojen objekt v Oraclovi bazi, katerega značilnost je, da pri vsakem klicu dodeli novo številko, ki

je ob privzeti uporabi za eno večja od prejšnje. Nad omenjenimi tremi tabelami sem postavil prožilce, katerih naloga je bila avtomatsko shranjevanje vrednosti sekvenc ob polnjenju dimenzij. V vsaki dimenziji sem nad poljem, ki je predstavljalo unikatno identifikacijo zapisa, postavil primarni ključ. Nad ostalimi polji pa sem glede na pričakovano distribucijo podatkov postavil B-drevesne (npr. KOLEDAR.MESEC) oziroma binarne indekse (npr. ORG_ENOTA.TIP).

Izgradnja tabele dejstev

Implementacija logične tabele dejstev RACUNI je bila izvedena na naslednji način:

V tablespace prostoru sem zgradil dve tabeli, RACUNI_DANES in RACUNI_ARHIV, obe z enako fizično strukturo, pri čemer sem uporabil particioniranje po razponih. Definiciji obeh tabel sem dodal oznako za paralelno izvajanje in zgoščevanje podatkov. Ker sem želel enoten pogled na obe tabeli, sem tabeli združil preko pogleda RACUNI z naslednjo definicijo:

```
CREATE VIEW racuni AS
SELECT * FROM racuni_danes UNION ALL SELECT * FROM racuni_arhiv;
```

Glavna razloga obstoja dveh tabel sta se nahajala v načinu osveževanja materializiranih pogledov in polnjenju podatkovnega skladišča. Materializirani pogledi potrebujejo za svoje osveževanje več časa, kot traja cikel polnjenja prototipnega podatkovnega skladišča. Zaradi tega bi se v praksi dogajalo to, da bi se materializirani pogledi stalno osveževali, uporabljali pa nikoli. Probleme polnjenja in z njim povezanega zaklepanja sem opisal v poglavju 3.4.2.2.

Nad polji, ki so predstavljala povezovalna polja z dimenzijami (npr. KUPEC_ID), sem postavil tuje ključe do dimenzij in na ta način vzpostavil referenčno integriteto. Vzpostavitev referenčne integritete je bila nujna, ker Oraclov optimizator poizvedb preko nje ugotovi, če ima opravka z dimenzijskim modelom in temu ustrezno prilagodi izvajanje poizvedb.

Sledila je postavitev indeksov. Pri tem sem poizkušal držati ugotovitev, opisanih v poglavju 3.4.2.2. Nad obe tabeli sem za potrebe operativnih poizvedb nad poljem KUPEC_ID postavil B-drevesni indeks. Indekse sem postavil še na določenih poljih tabele RACUNI_ARHIV, pri čemer je bil tip indeksa odvisen od pričakovane distribucije podatkov. Tako sem nad polji DATUM_IZDAJE, DATUM_ZAPADLOSTI in ORG_ENOTA_ID postavil binarni, nad ARTIKEL_ID in STEVILKA_DOKUMENTA pa B-drevesni indeks.

Na koncu sem nad tabelo RACUNI_ARHIV postavil še materializirani pogled. Z njim sem odstranil v poglavju 4.3.2.2 omenjeno slabost ROLAP izvedbe podatkovnega skladišča, počasnost izvajanja poizvedb. Njegova glavna uporabnost se je izkazala pri strateških poizvedbah, saj sem z njim pohitril poizvedbe nad arhivsko tabelo dejstev. Fizična struktura materializiranega pogleda se je od primarne

tabele razlikovala v tem, da ni vsebovala polja KLIENT_ID in DATUM_ZAPADLOSTI. Ostali parametri tabele, pri čemer mislim na tip particioniranja, zgoščevanje in paralelizem, so ostali enaki. Tip materializiranega pogleda je bil izbran tako, da je omogočal polnjenje s prirastkom na zahtevo.

4.3.4.3 Implementacija polnjenja aktivnega podatkovnega skladišča

Oracle ponuja več načinov polnjenja podatkovnega skladišča. Pred verzijo baze 9i je bilo običajno, da se je za polnjenje uporabljalo ali skripte ali Oraclovo orodje ETL Warehouse Builder. Za obe rešitvi je bilo značilno, da sta se izvajali običajno v nočnem času, ko transakcijski sistemi niso beležili sprememb. V verziji 9i so se pojavili zametki kontinuiranega polnjenja (omenjenega v poglavju 3.3.4), ki je v verziji 10g dozorel za produkcijsko uporabo. Oracle rešuje probleme kontinuiranega polnjenja s pristopom lovljenja spremenjenih podatkov (angl. *Change Data Capture, CDC*) na opazovanem sistemu. Opazovani primer je v tem primeru transakcijska baza podatkov.

Change Data Capture (CDC)

Z uporabo pristopa CDC je možno beležiti vse spremembe na transakcijskih tabelah, ki so posledica operacij INSERT, UPDATE in DELETE. Spremembe se shranjujejo v začasne tabele. CDC deluje preko dveh vmesnikov, založnika (angl. *publisher*) in naročnikov (angl. *subscriber*). Založnik je običajno administrator baze. Njegova naloga je, da na tabelah, katerih podatki bodo polnili podatkovno skladišče, postavi postopek zajemanja podatkov. Tako se za vsako tabelo na izvornem sistemu postavi tabela sprememb (angl. *change table*). Naročnike pa predstavljajo različni programi glede na svoje potrebe, zajemajo podatke iz izvornih tabel. Vsak naročnik lahko vidi isto tabelo sprememb na svoj način, kar pomeni samo tiste kolone, na katere je naročen, in tiste vrstice, ki jih še ni obdelal. Oracle podpira dve implementaciji mehanizma CDC:

- **Sinhroni CDC**: Temelji na prožilcih, ki se sprožijo vsakokrat, ko se na izvorni tabeli spremeni podatek. V tabelo sprememb se beležijo stare, nove vrednost ali celo oboje (pri stavkih UPDATE). Podatek se v tabeli sprememb vidi samo takrat, ko je v primarni tabeli potrjen. Dobra lastnost pristopa je, da omogoča beleženje sprememb v realnem času, to pa ima svojo ceno v nekoliko večji obremenitvi transakcijskega sistema, kar lahko pri visoko obremenjenih sistemih predstavlja problem. Drugi problem predstavlja potreba po spreminjanju objektov na izvornem sistemu, ki se mnogokrat izkaže kot pravna ovira. Ponudniki aplikacij namreč pogosto garantirajo pravilnost delovanja njihovih sistemov le ob pogoju, da se na produkcijskih objektih ne izvajajo nikakršne spremembe.

- **Asinhroni CDC**: Bistvena razlika med sinhronim pristopom CDC je ta, da asinhroni ne vpliva na delovanje transakcijskega sistema. Spremenjeni podatki se glede na potrebe po hitrosti zajemanja črpajo na dva načina:
    - Pristop HOTLOG bere spremembe iz trenutnih dnevnikov za obnovo podatkov, od koder jih Oraclov proces Stream prenaša v tabelo sprememb, ki se nahaja na izvorni strani. Spremenjeni podatki tu čakajo na naročnika.
    - Pristop AUTOLOG bere spremembe arhivskih dnevnikov za obnovo podatkov, potem ko so ti že preneseni na rezervno lokacijo. Ta pristop se običajno uporablja, ko obstaja na drugi lokaciji delujoča kopija primarnega SUBP (Data Guard, poglavje 4.3.3.2) in se jo uporablja za potrebe podatkovnega skladiščenja. Tabela sprememb se tako nahaja na drugi (sekundarni) strani.

Založniki ne glede na izbran pristop uporabljajo za potrebe generiranja tabele sprememb skupino ukazov, ki se nahajajo v paketu DBMS_CDC_PUBLISH, ki je del jezika PL/SQL.

Naročniki podatkov vidijo tabelo sprememb kot okno spremenjenih podatkov (angl. *queue*). Vsak cikel zajemanja podatkov je razdeljen na tri podkorake:

- **Povečanje okna**. Naročniku se zagotovi branje podatkov v tabeli sprememb, ki so prispeli do tega trenutka. Podatki, ki prispejo po povečanju okna, so vidni šele v drugem povečanju. Na ta način se zagotovi transakcijska pravilnost podatkov.
- **Branje podatkov**. Naročnik s pomočjo običajnega stavka SELECT bere podatke, ki so vidni v trenutnem oknu.
- **Čiščenje okna**. Ko so podatki v trenutnem oknu obdelani, se jih pobriše. Na ta način se zagotovi možnost naslednjega branja okna. Čeprav določen naročnik pobriše določeno okno, so podatki še vedno shranjeni v bazi vse dotlej, dokler ga ne pobrišejo vsi naročniki. Takrat se podatek dokončno pobriše iz začasne tabele.

Implementacija na prototipu

Konfiguracija sistema RAC je predvidevala skupno bazo podatkov, do katere dostopajo štirje strežniki. Zaradi pomanjkljivost sinhronega pristopa CDC sem se odločil za asinhroni pristop CDC. Zaradi skupne baze in relativno majhnih dovoljenih zakasnitvah sem izbral varianto HOTLOG. Proces polnjenja v skupno bazo podatkov je bil implementiral na strežniku za izvajanje strateških poizvedb (prvi v paru strežnikov, namenjen podatkovnemu skladiščenju). Sestavljali sta ga dve skripti:

- skripta za polnjenje operativnih podatkov, ki se je prožila vsakih 10 minut,

- skripta za prenos operativnih med strateške podatke, ki se je prožila vsako noč ob 2. uri.

Naloga skripte za polnjenje operativnih podatkov je bilo zagotavljanje čim hitrejšega polnjenja podatkovnega skladišča. Glavni koraki skripte so bili naslednji:

- Izvedi pomik vseh oken CDC, ki služijo kot vir podatkov.
- Osvežitev dimenzijskih tabel. Iz začasnih tabel CDC osveži dimenzijske tabele. Kjer je možno, za primarni ključ uporabljaj izpeljana numerična ali datumska polja. Postopek sem omenil v poglavju 4.3.4.1.
- Polni tabelo dejstev. Zajem podatkov za polnjenje se vrši iz dveh začasnih tabel CDC na produkcijski strani, ki vsebujeta podatke o naročilih in njihovih postavkah. Zajem se vrši s pomočjo enega stavka SQL, ki z ustreznimi transformacijami na poljih zagotovi kvaliteto podatkov. Stavek SQL istočasno poveže obe začasni tabeli CDC na produkcijski shemi in vse dimenzijske tabele v shemi podatkovnega skladišča s pomočjo ključev iz primarnega okolja. Spodnji stavek kaže del celotnega stavka.

  ```
  INSERT INTO racuni_danes
  SELECT ko1.datum, ko2.datum, p.stevilka_dokumenta, ku.kupec_id,...
  FROM prod.narocila_cdc p, prod.narocila_spec_cdc ps, dw.koledar
  ko1, dw.koledar ko2, dw.kupec ku, dw.org_enota o, dw. artikel a
  WHERE p.stevilka_dokumenta=ps.stevilka_dokumenta
  AND p.datum_izdaje=ko1.datum_izdaje AND p.kupec=ku.kupec ...
  ```

- Potrjevanje transakcije in čiščenje oken CDC. Po uspešnem ažuriranju dimenzij in dodajanju zapisov v tabelo dejstev skripta izvrši ukaz COMMIT in čiščenje trenutnih podatkov v oknih CSC.

Naloga druge skripte je bil prenos podatkov preteklega dne iz tabele RACUNI_DANES v RACUNI_ARHIV in osveževanje agregatne tabele. Skripta je morala zagotoviti transakcijsko pravilnost, ker bi se v drugačnem primeru lahko zgodilo dvojno štetje zapisov.

```
INSERT INTO racuni_arhiv SELECT * FROM racuni_danes
WHERE datum_izdaje < trunc(SYSDATE);
DELETE racuni_danes WHERE datum_izdaje < trunc(SYSDATE);
COMMIT;
```

Materializirani pogled se je osveževal po izvršenem prenosu v zgodovinsko tabelo dejstev. Ko so bili vsi podatki preneseni, se je izvedlo osveževanje statistik na vseh dimenzijskih tabelah, na zadnji particiji zgodovinske tabele dejstev ter materializiranega pogleda.

4.4 POVZETEK ANALIZE

4.4.1 Tehnološki vidik

Z vidika upravljanja z informacijami, je prototip internetne trgovine predstavljal kombinacijo informacijske demokracije in informacijske ambasade (podrobneje opisano v poglavju 2.2). Informacijska demokracija se je kazala z zagotavljanjem podatkov za potrebe notranjih analiz, informacijska ambasada pa s podatki o preteklih prodajah, ki so bili preko interneta izpostavljeni navzven v procesu dinamičnega izračunavanja popustov.

Implementacija prototipa je ustrezala zahtevam operativne poslovne inteligence, podanih v tabeli 1, saj je omogočala uporabo zgodovinskih informacij v aktivnem podatkovnem skladišču za potrebe operativnega odločanja z največ 10 minutnim zaostankom od trenutka, ko se je dogodek zgodil. Kot tako jo zato lahko uvrstim v fazo operacionalizacije, četrto fazo evolucije podatkovnega skladišča (faza je podrobneje opisana na strani 24).

Primerjava prototipa s tehnološkimi zahtevami aktivnih podatkovnih skladišč, podanimi v poglavju 3.4, je po posameznih kriterijih dala naslednje rezultate:

- **Zagotavljanje zmogljivosti**: Prototip je zagotavljal kombinacijo proaktivnega upravljanja poizvedb in dinamičnega dodeljevanja virov (glej 3.4.1.1). RAC konfiguracija je omogočala paralelizem z uporabo gruče večprocesorskih računalnikov (glej 3.4.1.3). S tem, ko se je posameznim tipom poizvedb dodelil določen strežnik, se je povečala zmogljivosti sistema. Zahteve zagotavljanja zmogljivosti na nivoju podatkovnega modela (glej 3.4.1.2) pa so bili izpolnjeni z indeksi, materializiranimi pogledi, zgoščevanjem podatkov ter spremenjeno velikostjo blokov na diskih.
- **Zagotavljanje razpoložljivosti in zanesljivosti:** Prototip je z RAC in Data Guard konfiguracijo izpolnjeval priporočila, ki so bila v poglavju 3.4.2 podana glede razpoložljivosti in zanesljivosti sistema. Prototip je podpiral visoko razpoložljivost tudi v primeru načrtovanih in nenačrtovanih izpadov sistema. Vpliv strojnih napak na procesnem delu je bil zmanjšan z gručo večprocesorskih računalnikov, na podatkovnem nivoju pa s SAN diskovnim poljem. Najzahtevnejša je bila implementacija Data Guard sistema, s katerim se je zagotovila razpoložljivost sistema v primeru katastrof.
- **Podatkovna ažurnost:** Prototip je podpiral kontinuirano polnjenje (glej 3.3.4), najnaprednejšo obliko polnjenja podatkovnih skladišč, in je kot tak v celoti izpolnil pričakovanja.

4.4.2 Stroškovni vidik

Prototip je bil namenjen predvsem analizi poslovnih in tehnoloških zahtev pri implementaciji aktivnega podatkovnega skladišča, zato stroškovna komponenta ni imela posebne teže. Izvedba prototipa je uporabljala obstoječo infrastrukturo in testne licence. Če bi želel prototip preslikati v produkcijsko okolje, bi se pri tem srečal z naslednjimi stroški:

- **Stroški SUBP**: Oracle ponuja dve obliki licenciranja. Prva se nanaša na število uporabnikov, druga pa na število procesorjev. Cena na uporabnika znaša 1.000 $, kar pri 6 računalnikih znaša minimalno 6.000 $. Na prvi pogled izgleda malo, vendar se lahko zgodi, da število uporabnikov močno naraste, tako da se izkaže ugodnejše licenciranje po posameznem procesorju (70.000$), kar v opisni konfiguraciji nanese okoli 400.000 $ (Oracle, 2007).
- **Stroški diskovnih polj**: Cene diskovnih polj se tako kot cene ostale strojne opreme nenehno nižajo. Ker gre pri njih večinoma za modularno izvedbo (z možnostjo nadgradenj), se lahko osnovne verzije diskovnih polj dobi za od 20.000 € dalje. To bi na začetku tudi zadostovalo za opisani prototip.
- **Stroški računalnikov in licenc za operacijski sistem**: Opisana konfiguracija bi na začetku delovala na strežnikih s ceno okoli 7.000 €, kar bi v celoti naneslo okoli 42.000 €.
- **Stroški postavitve podatkovnega skladišča**: Po podatkih na trgu se cene podobnih implementacij gibljejo od 50.000 € dalje.

Analiza stroškov pokaže, da postavitev aktivnega podatkovnega skladišča ni poceni ter pred uvedbo vsekakor zahteva analizo koristi in časa amortiziranja, ki bi ga implementacija sistema prinesla.

4.4.3 Končna ocena

V tabeli 5 za vsako od uporabljenih funkcionalnosti SUBP Oracle 10g prilagam še subjektivno oceno, podano na podlagi izdelave in preučevanja delovanja prototipa.

Zaključek, ki ga lahko podam na podlagi rezultatov, pridobljenih iz testiranj in analize prototipa, je, da lahko podjetja, ki se odločajo za implementacijo aktivnega podatkovnega skladišča, obravnavajo SUBP Oracle 10g kot resnega kandidata za izbiro. S tem se tudi potrjuje pravilnost na začetku tega poglavja omenjenih analiz, ki sta služili kot osnova za izbiro platforme, na kateri je bil izdelan prototip.

Tabela 5: Pregled uporabljenih funkcionalnosti, s prednostmi in slabostmi, pri implementaciji prototipa aktivnega podatkovnega skladišča s SUBP Oracle 10g

Segment	Funkcionalnost	Ocena
Strojna infrastruktura	RAC	+ povečana razpoložljivost sistema + možnost dodeljevanja virov + omogočanje paralelizma + sistem je razširljiv – cena implementacije (dodatni strežniki + licence RAC)
	Data Guard	+ zagotovljena zaščita sistema v primeru katastrof + vzdrževalni posegi ne obremenjujejo sistema – visoki stroški podvojitve primarne strani
Postavitev fizičnega modela	Particioniranje	+ preprostejše izvajanje administratorskih opravil na bazi + hitrejše poizvedbe – zahtevnejše začetno načrtovanje objektov
	Zgoščevanje objektov	+ prihranek na diskovnem prostoru + hitrejše poizvedbe – počasnejše dodajanje zapisov
	Paralelizem	+ hitrejše izvajanje operacij
	Bitmap indeksi	+ prihranek na diskovnem prostoru + optimizacija izvajanja poizvedb in s tem povečanje hitrosti izvajanja – počasnejše dodajanje zapisov
	Materializirani pogledi	+ hitrejše poizvedbe – potreba po dodatnem prostoru
	Delitev tabele dejstev	+ lažja kasnejša administracija + hitrejše polnjenje podatkov + optimalno zgoščevanje podatkov
Implementacija polnjenja	Asinhroni hotlog CDC	+ omogočanje kontinuiranega polnjenja + ne obremenjuje transakcijskega sistema – zahtevna implementacija polnjenja

Vir: Avtorjevo lastno delo

5 SKLEP

Operativna poslovna inteligenca postaja realnost. Tako kot danes povprečni uporabnik na urejevalnik teksta gleda kot orodje, bo v prihodnosti operativna poslovna inteligenca neopazno prisotna na ravneh, kjer bodo organizacije poizkušale izboljšati svoje operativno poslovanje. Sinergijski učinki se bodo navzven kazali kot izboljšan odnos z zunanjim svetom, povečanje kakovosti izdelkov in storitev, znižanje cene ter hitrejši odzivni čas. Vse to s ciljem čim bolj optimalnega poslovanje in dolgoročnega povečevanja dobička.

Operativna poslovna inteligenca ni nič drugega kot logična evolucija tradicionalne poslovne inteligence. S tem ko je z njo omogočeno odločanje na vseh ravneh organizacije, je operativna poslovna inteligenca postala kritičen del informacijskih sistemov v organizaciji. Pod kritično je mišljena njena tehnična komponenta, katere najpomembnejši del predstavlja podatkovno skladišče, ki predstavlja bazo podatkov.

Čeprav teorija operativne poslovne inteligence obstaja že nekaj let, je preslikavo teoretičnih konceptov v realnost omejevala nedozorela tehnologija, predvsem nepopolnost sistemov za upravljanje baz podatkov. Ti so bili v začetnih fazah svojega razvoja orientirani v zagotavljanje podpore transakcijskim informacijskim sistemom, delno tudi tradicionalnem podatkovnem skladiščenju. Nekateri ponudniki SUBP so v zadnjem obdobju ugotovili, da bodo morali podpreti tudi operativno poslovno inteligenco, kot najmlajši koncept poslovne inteligence. Nove funkcionalnosti so evolucijsko vgrajevali v obstoječe SUBP. Sedanje verzije določenih SUBP so izpopolnjene do te mere, da so lahko uporabne za shranjevanje transakcijskih podatkov in tudi za potrebe aktivnega podatkovnega skladiščenja. Gre za združitev dveh svetov: transakcijskega, za katerega so značilne hitre poizvedbe, neprekinjeno delovanje sistemov in velike količine uporabnikov, ter analitičnega, katerega značilnost so velike količine podatkov, nepredvidene in analitično zahtevne poizvedbe.

Ravno združevanje obeh svetov je verjetno razlog, da se bo v bodočnosti verjetno, po zadnjih raziskavah (Feinberg, 2006; Gartner, 2006), boj za prevlado na trgu odvijal med ponudniki SUBP, ki temeljijo na paralelnih računalnikih (Teradata), in ponudniki SUBP, delujočih na večprocesorskih računalnikih (IBM, Oracle, Microsoft). Ker je vzdrževanje takšnih sistemov zahtevno in drago, so se v zadnjih petih letih pričeli pojavljati namenski sistemi za podatkovno skladiščenje. Glavna proizvajalca teh sistemov sta podjetji Netezza in DATAllegro. Oba ponujata sisteme, sestavljene iz modulov, ki so si med seboj enaki. Sistemi imajo neomejeno možnost nadgradnje z novimi moduli. Čeprav ti sistemi nimajo dolge zgodovine, obstaja verjetnost, da bodo v prihodnosti, zaradi preproste

administracije, ugodne cene in dobrih zmogljivosti, postali pomemben udeleženec na trgu ponudnikov sistemov za podatkovno skladiščenje.

Zaradi majhne razširjenosti sistemov za operativno poslovno inteligenco sem se odločil njene koncepte preveriti v praksi, na čisto življenjskem primeru. Skozi celotno izdelavo magistrskega dela pa sem se vseskozi srečeval s tehnološko zahtevnostjo izvedbe in visokimi stroški, s katerimi bi se srečal v primeru produkcijske postavitve sistema. Ugotovil sem, da je pri trenutnem stanju potrebno mnogo interdisciplinarnega znanja, ki obsega področja:

- sistemske administracije (računalnikov,diskovnih polj in mrežnega okolja),
- poznavanja specifik posameznih SUBP (na administratorskem in razvojnem nivoju),
- poznavanja relacijskega in dimenzionalnega modeliranja ter
- poslovnih znanj, potrebnih za ugotavljanje in razumevanje uporabniških potreb.

Na srečo je informatika eno izmed področij, v katerih se odvijajo hitre tehnološke, vsebinske in cenovne spremembe, zato mislim, da bodo sistemi operativne poslovne inteligence v nekaj letih postali realnost tudi v branžah, ki si danes tako zahtevnih sistemov ne morejo privoščiti.

Lahko zaključim, da je prototipna izdelava sistema operativne poslovne inteligence pokazala, da je v relativno kratkem časovnem obdobju mogoče priti do v praksi delujočega sistema. S tem sem se prepričal, da je današnja tehnologija pripravljena na izzive, ki jih ponuja operativna poslovna inteligenca, s čimer so bili izpolnjeni zastavljeni cilji, podani v uvodu magistrskega dela.

6 LITERATURA IN VIRI

6.1 LITERATURA

1. Agosta, Lou: Data Warehousing Practioners Favor Hub-And-Spoke Architecture, Forrester Research, 2004, 4 str.
2. Agosta, Lou: The Data Strategy Advisor: Is your Data Warehouse Active?, DM Review, april 2005
3. Akbay Sami: Data Warehousing in Real Time, Business Intelligence Journal, 2006, številka 11, 22-28 str.
4. Ankorion Itamar: Change DataCapture – Efficient ETL for Real-Time BI, DM Review, januar 2005
5. Arvey Richard et al.: Mainstream Science on Intelligence, WallStreet Journal, 13.12.2004
6. Atre Shaku: Operational Business Intelligence Applications: Business Case Assessment, [URL:http://www.b-eye-network.com/view/2529], 1.9.2006
7. Atre Shaku: Operational Business Intelligence Applications: Infrastructure is the Key to Success, [URL:http://www.b-eye-network.com/view/2849], 1.9.2006
8. Atre Shaku: Two Main Pillars of Operational Business Intelligence, [URL:http://www.b-eye-network.com/view/3100], 1.9.2006
9. Barb Lundhild: Oracle RAC 10g, [URL:// http://www.oracle.com/technology/ products/database/clustering/pdf/twp_rac10gr2.pdf], 10.11.2006
10. Basu Raj: Challenges of Real-Time Data Warehousing, DW Review, julij 2003, 9 str.
11. Bazerman Max: Judgment in Managerial Decision Making, Wiley, 2001, 192 str.
12. Berry Michael, Linoff Gordon: Mastering Data Mining The Art and Science of Customer Relationship Management:, John Wiley & Sons, 2002, 494 str.
13. Berson Alex, Kurt Thearling: Building Data Mining Applications for CRM, Mc-Graw Hill, 1999, 488 str.
14. Biere Mike: Business Intelligence for the Enterprise, IBM Press, 2003, 240 str.
15. Blansfield David: The First Word, [URL: http://www.bpmmag.net/magazine/ article.html?articleID=13966], 6.2.2007
16. Brobst Stephen: Designing a High-Performance Data Warehouse, TDWI, 2006, 287 str.

17. Brobst Stephen: Real-Time Data Warehousing, TDWI, 2006, 301 str.
18. Brobst Stephen, Rarey Joe: The Five Stages of an Active Data Warehouse Evolution, Teradata Magazine, 2001, 38-44 str.
19. Chyran Michele: Oracle Database Concepts 10g Relase 2, Oracle, 2005, 548 str.
20. Drobik Alexander et al.: The Gartner Definition of Real-Time Enterprise, COM-18-3057, Gartner Group, 2002
21. El-Nasr Magy: Agents, Yen John, Emotional Intelligence and Fuzzy Logic, Texas A&M University, 1998, 118 str.
22. Feinberg Donald, Beyer Mark A.: Magic Quadrant for Data Warehouse Database Management Systems, Gartner Group, 2006
23. Freeman Robert G: Oracle Database 10g New Features, The McGraw-Hill Companies, 2004, 233 str.
24. Gartner Group: Gartner Says Worldwide Relational Database Market Increased 8 Percent in 2005 [URL:http://www.gartner.com/press_releases/asset_152619_11.html], 10.10.2006
25. Gilmore, James H.: Markets of One, Harvard Business School Press, 2000, 240 str.
26. Gold-Bernstein Beth: EAI Market segmentation, EAI Journal, julij/avgust, 1999
27. Graefe Goetz, Shapiro Leonard D.: Data Compression and Database Performance, ACM/IEEE-Computer Science Symposium, 1991, 10 str.
28. Hackathorn Richard: The BI Watch Little BI Versus Big BI, DM Review, januar 2001, [URL: http://www.dmreview.com/article_sub.cfm?articleId=2908]
29. Hackathorn Richard: What is Active Business Intelligence, DM Review, maj-junij 2002
30. Hager Vincent: Active DataWarehousing in The Real-Time Enterprise, [URL:http://www.big.tuwien.ac.at/teaching/offer/ss05/usi1/usi1adw.pdf#search=%22active%20datawarehousing%20teradata%20hager%22], 1.9.2006
31. Hall Curt: Business Process Intelligence, Business Process Trends, junij 2006, 9 str.
32. Hobbs Lilian et al.: Oracle 10g Data Warehousing, Digital Press, 2004, 872 str.
33. Inmon Bill: The Operational Data Store, InfoDB, februar 1995, 4 str.
34. Inmon Bill: Building the Data Warehouse, Fourth Edition, Wiley Publishing, 2005, 578 str.
35. Kimball Ralph et al.: The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling, Wiley, 2002, 464 str.
36. Klein Gary, Orasanu Judith, Calderwood Roberta: Decision Making in Action: Models and Methods, Ablex Publishing, 1993, 448 str.

37. Lane Paul: Data Warehousing Guide, Oracle Corporation, 2002, 666 str.
38. Liautaud Bernard, Hammond Mark: E-Business Intelligence, McGraw-Hill, 2001, 306 str.
39. Loshin David: Business Intelligence: The Savvy Manager's Guide, Morgan Kaufmann Publishers, 2003, 253 str.
40. Ma Catherine, Chou C. David, Yen C. David: Data warehousing, technology assessment and management, Industrial Management & Data Systems, April 2000, 125 – 135 str.
41. Nickels William: Understanding Business, MC Graw-Hill, 1999,768 str.
42. Ozbutun Cetin: Table Compression in Oracle Database Release 2, Oracle Corporation, 2005, 8 str.
43. Parida Rajeev A.: Principles and Implementation of Data Warehousing, Firewall Media, 2006, 483 str.
44. Paul Seth, Gautam Nitin, Balint Raymond: Preparing and Mining Data eith Microsoft SQL Server 2000 and Analysis Services, Microsoft Corporation, 2002, 154 str.
45. Pendse Nigel: What is OLAP?, OLAP Report, [URL:http://www.olapreport.com/fasmi.htm], 10.11.2006
46. Piatetsky-Saphiro Gregory, Frawley William: Knowledge Discovery in Databases, AAAI Press/MIT Press, 1991, 539 str.
47. Puttanqunta Naveen: Active Data Warehousing and the Quest for Real-Time BI, DM Direct Newsletter, oktober 2006, [URL: http://www.dmreview.com/article_sub.cfm?articleID=1065026]
48. Rosenberg Alex: Improving Query Performance in Data Warehouses, Business Intelligence Journal, 2006, št. 11, 7-12 str.
49. Scalzo Bert: Oracle DBA Guide to Data Warehousing and Star Schemas, Prentice Hall, 2003, 240 str.
50. Schupmann Vivian: Oracle Data Guard Concepts and Administration, Oracle Press, 2006, 386 str.
51. Theodoratos D., Sellis T.: *Designing Data Warehouses*, Data & Knowledge Engineering, November, 1999.
52. Two Crows Consulting [URL: http://www.twocrows.com], 1.12.2006
53. Vitt Elizabeth, Luckevich Michael, Misner Stacia: Business Intelligence, Microsoft, 2002, 192 str.
54. White Colin: Data Integration: Using ETL, EAI, and EII Tools to Create an Integrated Enterprise. TDWI – What Works, 2006, številka 21
55. Yan, An integrated high availability computing platform, The Electronic Library, 2005, 632–640 str.

56. Yoles Maurice, Understanding the Inteligent Organization: Organizacija, 2006, številka 1, 11 str.

6.2 VIRI

1. California State University Monterey Bay, Data Warehouse Glossary. [URL:http://it.csumb.edu/departments/data/glossary.html], 13.9.2006
2. DBA Support, [URL:http://www.dbasupport.com], 10.10.2006
3. GoldenGate Software [URL:http://www.goldengate.com/solutions/rtdataware.html], 8.1.2007
4. ITToolbox, [URL:http://www.ittoolbox.com], 2006
5. Kimball Ralph: Podatkovno skladiščenje - interno izobraževalno gradivo, Bled, 1999
6. Oracle Metalink, [URL:http://metalink.oracle.com], 2007
7. Oxford Reference, Oxford University Press, Elektronska izdaja, 2006
8. Slovar informatike, [URL:http://www.islovar.org], 2006
9. Sybase Techwave: Zapiski iz uporabniške konference, Orlando, 1999
10. TDWI, [URL:http://www.tdwi.org], 2006
11. University of California Santa Cruz, Data Warehouse Glossary. [URL:http://planning.ucsc.edu/irps/dwh/DWHGLOSS.HTM], 13.9.2006
12. Weibull,[URL:http://www.weibull.com/SystemRelWeb/basic_system_reliability_terminology.htm], 10.1.2007
13. Weik Martin: Communications Standard Dictionary, Chapmann & Hall, 1996, 1185 str.
14. Wikipedia, Business Intelligence. [URL: http://en.wikipedia.org/wiki/Business_intelligence], 13.9.2006
15. Yahoo Finance, [URL: http://finance.yahoo.com/q/pr?s=ORCL], 14.1.2007

7 SLOVAR SLOVENSKIH PREVODOV TUJIH IZRAZOV

Archive log	dnevnik sprememb
Bitmap Index	binarni indeks
Buffer	medpomnilnik
Business Performance Management (BPM)	management uspešnosti in učinkovitosti
Business to Business (B2B)	medpodjetniško poslovanje
Changed Data Capture (CDC)	princip polnjenja podatkovnega skladišča, ki temelji na zajemu spremenjenih podatkov
Closed Loop Processing	procesiranje s povratno zanko
Clustering, cluster	gručenje, gruča
Cold backup	način izdelovanja varnostnih kopij med nedelovanjem baze podatkov
Control file	kontrolna datoteka
Cost Based Optimization (CBO)	način optimizacije poizvedb glede na njihovo zahtevnost
Dashboard	nadzorna plošča
Data Dictionary	podatkovni slovar, repozitorij
Data file	podatkovna datoteka
Data Mart	področno podatkovno skladišče
Data Mining (DM)	podatkovno rudarjenje
Dedicated	individualno namenjen, posvečen
Dirty read	branje še nepotrjenih podatkov
Dynamic Partition Pruning	dinamično omejevanje particij
Enterprise Application Integration (EAI)	integracija aplikacij
Extract Transform Load (ETL)	proces polnjenja podatkovnega skladišča
Fail over	funkcionalnost, ki omogoča prehod izvajanja operacij iz enega na drugi

	strežnik
Fault tolerant system	sistem, ki je varen pred izpadi zaradi katastrof
Fraud detection	odkrivanje zlorab
Full Table Scan	branje celotnih tabel
Grid	mreža
Hot backup	način izdelovanja varnostnih kopij med delovanjem baze podatkov
Join	povezava, zveza
Key Performance Indicators (KPI)	sistem uravnoteženih kazalnikov
Link analysis	povezovalna pravila
Massively Parallel Processing (MPP)	konfiguracija računalniškega sistema z uporabo paralelnih računalnikov
Network Attached Storage (NAS)	omrežno priključene diskovne kapacitete
OnLine Analytical Processing (OLAP), ROLAP HOLAP MOLAP	analitična obdelava podatkov relacijska implementacija OLAP-a hibridna implementacija OLAP-a multidimenzionalna implementacija OLAP-a
Object Request Broker (ORB)	protokol za komunikacijo med objekti
On Commit	po potrditvi
On Demand	na zahtevo
Operational Data Store (ODS)	operativna podatkovna hramba
Pivot	vrtenje
Plan-Do-Check-Act (PDCA).	Planiraj-Izvedi-Preveri-Ukrepaj
Program Global Area (PGA)	spomin posameznih procesov
Pull	vlečenje, zajemanje
Push	potiskanje
Query rewrite	postopek, ki primarno poizvedbo prepiše v optimalno poizvedbo

Recovery	obnovitev, povrnitev
Redo log file	dnevnik za obnovo podatkov
Redundant Array of Independent Disks (RAID)	način varovanja podatkov z redundantno skupino neodvisnih diskov
Replication	podvojitev
Query optimizer	optimizator poizvedb
Real Application Cluster (RAC)	gruča računalnikov, ki dostopa do iste podatkovne baze
Remote Procedure Call (RPC)	klic za oddaljeni postopek
Scheduled job	urnik izvajanja
Shared	v skupni rabi, skupen
Storage Area Network (SAN)	samostojno diskovno polje
Symetric Multi Processing (SMP)	konfiguracija računalniškega sistema z uporabo gruče večprocesorskih računalnikov
System Global Area (SGA)	spomin sistema
Total Cost of Ownership (TCO)	stroški lastništva
Transaction Data Movement (TDM)	kontinuirano polnjenje
Transpose	transponiranje
Trigger	prožilec
View	pogled (mišljena je struktura v relacijskih bazah)
Workflow	delovni tok