Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Diplomová práce

DBS ORACLE a jeho použití v rámci úřadu veřejné správy

Bc. Michal Král

Zadávací list

Čestné prohlášení

Prohlašuji, že svou diplomovou práci "DBS ORACLE a jeho použití v rámci úřadu veřejné správy" jsem vypracoval samostatně pod vedením vedoucí diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu literatury na konci práce. Jako autor uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 10. dubna 2009 _____

Poděkování

Úvodem mé práce děkuji paní RNDr. Evě Jablonské, CSc., za odborné vedení, rady a připomínky k diplomové práci.

Dále děkuji redaktoru internetového portálu panu Ing. Marku Kocanovi za poskytnutí materiálů, informací a podkladů pro vznik této práce. Poděkování patří i panu Michalu Hrochovi ze společnosti Microsoft©, který rovněž svými informacemi napomohl ke zkvalitnění této práce. A poslední poděkování patří panu Ing. Pavlu Holubovi za věcné a i odborné připomínky k této práci.

DBS ORACLE a jeho použití v rámci úřadu veřejné správy

DBS ORACLE and its implementation within public administration offices

Souhrn

Tématem diplomové práce je relační databázový systém Oracle a jeho implementace v rámci úřadu veřejné správy. Práce se však nevěnuje pouze tomuto rozšířenému databázovému systému, ale jsou v ní představeny i další pro Oracle konkurenční systémy.

Velká pozornost je věnována problematice zálohování dat v databázi a jejich obnově v databázovém systému Oracle, jako důležitému kritériu výběru databázového prostředí v konkrétní organizaci veřejné správy. Další informace jsou zaměřeny na navrhnutí konkrétního optimálního řešení pro středně velký úřad veřejné správy.

Dílčím cílem práce je posoudit z různých hledisek (výkon, bezpečnost, stabilita), zda nasazení databázového systému Oracle je pro konkrétní úřad veřejné správy vhodnou volbou. Součástí práce je test zálohování i obnovy databázového systému, odhalující, na které hardwarové parametry je vhodné se při nákupu serveru zaměřit.

Hlavním cílem této práce je poskytnout metodické pokyny pracovníkům IT, kteří pracují na úřadech veřejné správy a ulehčit jim rozhodování, který databázový systém je pro jejich konkrétní úřad vhodný. Orientace mezi databázovými systémy je velmi obtížná a řada pracovníků úřadů, včetně informatiků, dává přednost tzv. „hotovým řešením“, která obsahují kromě vlastního databázového systému i nabídku aplikačního softwaru, který úřad zajímá nejvíce.

Klíčová slova

Databázový systém, Oracle, veřejná správa, řešení informačního systému

Summary

The thesis subject is relation database system Oracle and its implementation into public administration. The goal of this thesis is to focus not only on the Oracle but also its competition.

Great attention is paid to the issue of data backup in the database and restoring the Oracle database system, as important criteria for selecting the specific database environment in the organization of public administration. Further information is geared towards proposing a specific optimal solution for medium-sized office of public administration.

The main objective of this thesis is to assess the various aspects (performance, security, stability) in the deployment of Oracle DBS as an appropriate choice. The work includes the server hardware test, which reveals that with the hardware parameters, it is appropriate to the purchase of a server address.

The intention is to facilitate the work of IT administrators who work for public authorities, decision-making, what database system for their office is an appropriate choice. General awareness of the database systems is not extensive, because for many Informatics – Administrators nowadays it is not easy to orientate between database systems and trace the way that best suits the needs of the office.

Keywords

Database system, Oracle, Public administrativ, solution of information system

Obsah

1 Úvod

Každý promyšlený informační systém, který dnes používají nejen instituce veřejné správy, ale i komerční instituce, musí splňovat velké množství kritérií. Bezpečnost, stabilita, funkčnost nebo spolehlivost informačního systému bude záležet z velké části na zvoleném databázovém systému, který je základem každého informačního systému.

Databázový systém je pojem, který zahrnuje samotné údaje spravované v databázi společně se softwarem pro přístup k těmto údajům.

Databázi, lze chápat jako úložiště údajů, které jsou zpracovány a uloženy nezávisle na aplikačních programech. Databáze obsahují jednak vlastní údaje, ale i relační vztahy mezi jednotlivými elementy a objekty v databázi, schémata popisující struktury údajů a integritní omezení.

Pro přístup k datům uloženým v databázovém systému se využívá speciální software. Nazývá se anglicky Database Management System (DBMS), česky Systém řízení báze dat (SŘBD). Jak jsou údaje uložené fyzicky, nemusí být aplikačnímu programu, a tedy ani uživateli, známo.

Cílem této diplomové práce je hlouběji proniknout do problematiky databázových systémů a ukázat jejich použitelnost v instituci veřejné správy. Výstupem by pak měl být návod pro specialisty IT, kteří mají v kompetenci rozhodování o databázových systémech na svých úřadech, jak vybrat databázový systém, který by splňoval požadavky na něj kladené.

2 CÍL PRÁCE A METODIKA

Hlavním cílem práce je posoudit z různých hledisek (výkon, bezpečnost, stabilita), zda nasazení DBS[1] Oracle je pro konkrétní úřad veřejné správy vhodnou volbou. Zvláště je cíl práce zaměřen na zálohování a obnovu databáze.

Dílčím cílem této práce je usnadnit specialistům IT, kteří pracují na úřadech veřejné správy, rozhodování, který databázový systém je pro jejich úřad vhodný. Orientace mezi databázovými systémy je velmi obtížná a řada pracovníků úřadů, včetně specialistů IT, dává přednost tzv. „hotovým řešením“, která obsahují kromě vlastního databázového systému i nabídku aplikačního softwaru, který úřad zajímá nejvíce.

Z předchozího odstavce tedy vyplývají další parciální cíle této diplomové práce a to hlouběji proniknout do problematiky databázových systémů a ukázat použitelnost databázových systémů v instituci veřejné správy.

K naplnění cílů je třeba splnit následující body:

- Vymezit základní pojmy týkající se problematiky databázových systémů.
- Představit databázové systémy dostupné na českém trhu, které jsou relevantní pro úřad veřejné správy.
- Stanovit kritéria, na jejichž základě je vhodné se rozhodnout pro určitý databázový systém v úřadu veřejné správy.
- Následně představit databázový systém Oracle jako možnou volbu.
- Ukázat stěžejní bod databázových systémů – způsoby zálohování a obnovy dat v konkrétním databázovém prostředí Oracle.
- Ukázat konkrétní postup zálohy dat a následné obnovy v instituci Magistrátu města Pardubic.
- Navrhnout vhodné konkrétní řešení pro středně velký úřad veřejné správy.

[1] Databázový systém – zkráceně DBS.

V teoretické části byly použity metody:

- studium literatury,
- analýza administrátorských dokumentací,
- porovnávání poznatků (komparace),
- citace,
- deskripce (na základě studia z Oracle University),
- kompilace.

V praktické části byly použity metody:

- Rozhovory (s databázovými specialisty, administrátory, z úřadů a společností, kteří se problematikou databázových systémů zabývají).
- Pozorování (chování databázových systémů na různých hardwarových konfiguracích).
- Měření (zálohy a obnovy při změnách hardware).
- Analýza výsledků testů zálohy a obnovy).
- Vyhodnocení výsledků testů zálohy a obnovy.

Měření zálohy a obnovy

Měření bylo prováděno na databázovém systému Oracle verze 9.2.0.8. Jednotlivá měření byla rozdělena podle hardwaru, který prokazatelné ovlivňuje rychlost a spolehlivost obnovy. Na základě znalostí databázového systému Oracle bylo přistoupeno pouze k testování výměny paměťových modulů a pevného disku. V prvním srovnávacím testu byly ponechány výchozí parametry serveru[2]. Další test spočíval ve výměně pevného disku za disk s vyšším počtem otáček za minutu. Poslední srovnávací test byl založen na navýšení velikosti paměťového modulu na dvojnásobek. Celkem bylo provedeno 10 měření.

Výsledky měření jsou vyjádřeny konkrétními hodnotami.

[2] Kapitola vliv hardwaru na zálohování a obnovu - kapitola 5.5.5.

Cílem měření bylo zjistit, který databázový systém lze zálohovat a obnovit rychlejším způsobem, přičemž jsou testovány různé kombinace hardwaru a to vše za předpokladu úspěšné zálohy i obnovy s výslednými konzistentními daty. Velikost databáze je 20 GB.

3 DATABÁZOVÉ PLATFORMY NA ČESKÉM TRHU

Databázový systém lze charakterizovat jako soubor programů, které byly navrženy podle dohodnutých kritérií tak, aby soustředily data do jednoho celku, obhospodařovaly je a umožňovaly přístupy k různým aplikacím a různým uživatelům najednou [3]. Z historie vzešly tři hlavní databázové modely, a to hierarchický, síťový a relační. Nejstarší z uvedených je hierarchické modelování databází. Toto pojetí pochází z reálného uspořádání světa. Jako příklad může sloužit model organizace moci, rozklad výrobků na součástky, strom adresářů aj. Pro hierarchické modelování je typická práce se stromy, kdy ve stromu jsou realizovány vztahy 1:N. Příkladem hierarchické databáze může být IMS od firmy IBM či M-system od firmy Micronetics [18].

Variací hierarchického modelu je síťový model databáze. V síťovém modelování je možné vyjadřovat vedle vztahů 1:N i vztahy M:N. Fyzická realizace síťového modelu je ale náročná a aktualizace obvykle komplikovaná [18].

Základní výhodou hierarchického a síťového modelu je efektivnost zpracování, tj. rychlost přístupu k datovým záznamům. Na druhé straně mezi nevýhody patří obtížnost jednou nadefinované stromy a vazby mezi stromy měnit. Nejsou uzpůsobeny pro dotazy.

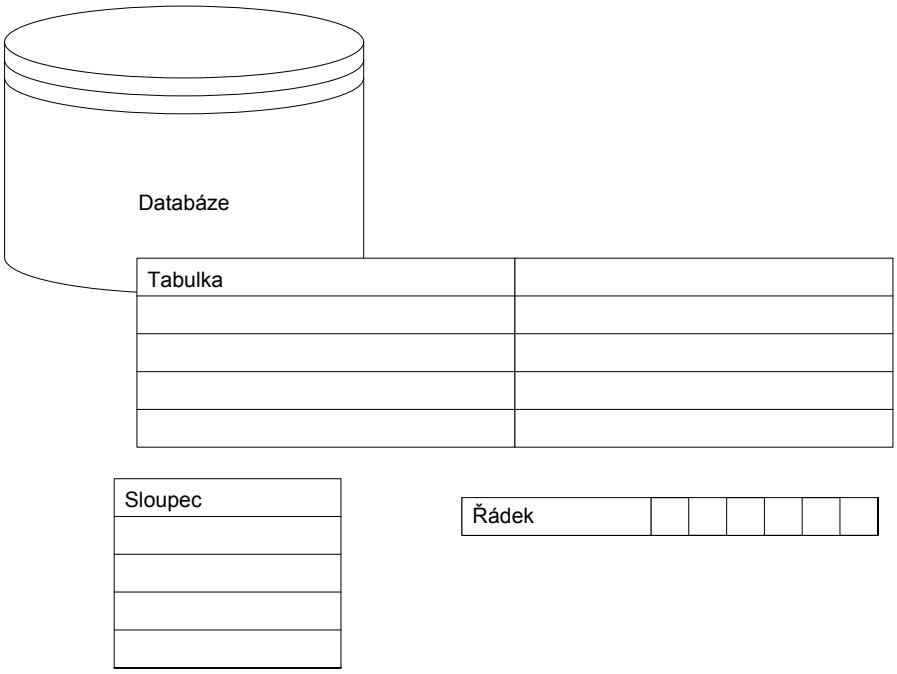
Z teoretického hlediska je nejpropracovanější relační model databáze, který byl vyvinutý doktorem E. F. Coddem už v šedesátých letech minulého století.

Relační model definuje způsob, jakým je možné reprezentovat strukturu dat, způsoby jejich ochrany a operace, které můžeme nad daty provádět. Relační databáze je sestavená z řady tabulek, jejichž sloupce jsou vázány na sloupce v jiných tabulkách. Takto propojená datová pole jsou na sobě určitým způsobem závislá. Jejich vztahy jsou založeny na klíčových hodnotách uložených v příslušných sloupcích.

U relačních databází je základní výhodou relativně snadná modifikace a propojování tabulek a s nimi spojená možnost dotazů. Slabým místem je nízká efektivnost zpracování, což se projevuje v tom, že řada příkazů vyžaduje velké množství přístupů na disk a tím se zpomaluje zpracování [18].

Tato práce se zabývá v současné době nejrozšířenějším modelem relačním. Pro úplnost je třeba dodat, že v současnosti je dále využíván objektový a objektově – relační přístup k databázím [18].

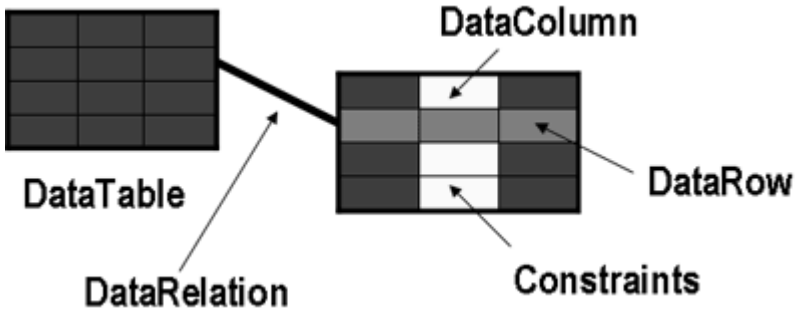
Na obrázku 1 je ukázka vztahu mezi databází, tabulkou, řádkem a sloupcem v relačních databázích:



Obrázek 1 - Databáze, tabulka, řádek, sloupec, zdroj: [vlastní].

Tabulku v databázi lze rovněž popsat pomocí různých atributů. Nejdůležitější údaje, které určují pozici dané hodnoty, jsou řádek (row) a sloupec (column). Důležitým údajem je rovněž tzv. constraint. Constraint má podobnou funkci jako index, může však být použit k nastavení relací s jinou tabulkou. Patří do kategorie podmínek.[3] Více obrázek 2.

[3] Microsoft Corporation. *Microsoft Office Online : Microsoft Office Access* [online]. Microsoft Corporation, 2009, 2009 [cit. 2009-01-12]. Dostupný z WWW: <http://office.microsoft.com/cs-cz/access/HA012314371029.aspx>.



Obrázek 2 - Struktura tabulky, zdroj: [4].

3.1 SQL PROGRAMOVACÍ JAZYK

S relačními databázovými systémy se samozřejmě začaly vyvíjet a postupně zdokonalovat dotazovací jazyky. V praxi se můžeme setkat se dvěma koncepty dotazovacích jazyků:

- koncept QBE[4],
- koncept SQL[5].

Při práci s QBE jde spíše o jednoduchý zápis dotazu do navrženého schématu (formuláře) než o "programování". Koncept QBE tak umožňuje rychle vytvářet dotazy i běžným uživatelům, nejen odborníkům. Původní vznik konceptu QBE byl spojen s firmou IBM už v 70. letech, později jej velmi kvalitně uplatnila firma Borland v produktu Paradox [20].

František Skřivánek[6] popisuje základy dotazovacího jazyka SQL. Jazyk SQL patří do kategorie neprocedurálních[7] jazyků, je možné vystopovat do první poloviny sedmdesátých let. V průběhu roku 1974 totiž vznikl ve vývojových laboratořích firmy IBM jazyk Sequel. Ten byl použit coby nosný jazyk systému R, který se tak de facto stal prvním systémem využívajícím jazyk SQL (některé zdroje dnes uvádějí, že takových databázových produktů většího významu je již více než stovka – těch opravdu významných pak několik desítek).

[4] Query By Example – dotazovací databázový jazyk pro relační databáze.
[5] Structured Query Language – neprocedurální dotazovací jazyk.
[6] František Skřivánek – redaktor web portálu Databázový svět – www.dbsvet.cz.
[7] Neuzavřený kód, pomocí příkazů říkáme, co chceme provést, nikoli jak to chceme provést [7].

Jazyk SQL byl postupně přijat jako standard různými výrobci databázových aplikací a stal se tak spojovacím článkem mezi různými systémy. V koncepci klient/server se dotazy specifikují na straně klienta, odesílají na stranu serveru, který dotaz v jazyce SQL realizuje a výsledek pošle zpět uživateli na straně klienta. Dnes většina významných databázových platforem jazyk SQL podporuje [20].

Vzrůstající význam relačních databází si vyžádal nutnost standardizace. A tak v roce 1986 byl Americkým standardizačním institutem (ANSI) přijat nový standard SQL-86. Tento standard měl však některé nedostatky, a tak byl v roce 1992 přijat standard SQL-92 (Označován též SQL2.). V současnosti je v platnosti norma SQL3, která reaguje na poslední trendy vývoje v databázových technologiích (např. využívání objektů apod.) [20].

Jazyk SQL můžeme částečně využít pro samotný vývoj databázových aplikací, ale jinak slouží především jako dotazovací jazyk pro práci s údaji v relační databázi.

Příkladem vyspělých SŘBD jsou produkty Oracle 10g Database, IBM DB2, MS SQL Server [14]. Přestože byl jazyk SQL od počátku koncipován jako dotazovací neprocedurální jazyk, v průběhu vývoje se objevilo mnoho procedurálních rozšíření. Mezi tato rozšíření patří především podpora práce s cykly, větvení běhu programu, funkcí a procedur. I přes existující standardy SQL jsou však tato rozšíření velmi závislá na implementaci v konkrétním systému. Tyto procedurální prvky jsou dnes součástí každého vyspělého databázového serveru – mezi známé implementace patří například Transact SQL či Oracle PL/SQL [20].

V poslední době se také stále více využívá SQL příkazů v rámci jiných programovacích jazyků – tato "vestavěná" forma jazyka SQL je tak vhodným doplňkem klasických procedurálních jazyků. Tento přístup je vhodný zejména v případě aplikací komunikujících interaktivně s uživatelem – jazyk SQL totiž nedisponuje takřka žádnými možnostmi pro tvorbu uživatelského rozhraní [20].

3.2 NEJROZŠÍŘENĚJŠÍ DATABÁZOVÉ SYSTÉMY

Mezi současné trendy v databázových technologiích patří především snaha o zajišťování vysoké dostupnosti a výkonu, možnosti rozložení zátěže a podpory

zpracování v gridu[8], snižování celkových nákladů na vlastnictví, zjednodušování administrace, zajištění přístupu k datům uloženým v heterogenních zdrojích či o automatickou správu a ladění. Stranou zájmu nezůstávají ani snahy o stále větší zabezpečení, i když na tuto oblast slyší jen určité procento zákazníků, znajících hodnotu svých dat.

Současné databázové systémy ve větší, či menší míře podporují jazykové zpracování textů, nejčastěji pak v podobě fulltextového vyhledávání, doplněné o některé významové prvky. Samozřejmostí se stává také zpracování dalších nestrukturovaných dat, včetně dat multimediálních.

Velký potenciál lze spatřovat i v podpoře technologie XML[9], ať již v podobě mapové, aplikované na některý současný model, tak i v podobě nativních XML databází. Stranou zájmu již dlouhá léta nezůstávají ani prostorová a geografická data, používaná například pro provoz geografických informačních systémů, stejně jako podpora třívrstvé i vícevrstvé architektury.

Stejně jako v každé jiné oblasti lidského snažení, probíhá i mezi dodavateli databázových technologií nelítostný souboj o mnohá „nej“ – od největšího podílu na trhu, přes nejrychlejší zpracování standardních i nestandardních situací, až po nejlepší databázovou platformu vůbec. Přitom zejména poslední uvedené „nej“ je nedosažitelné.

Pojem „nejlepší“ je v souvislosti s databázovými technologiemi zavádějící, přesněji nepoužitelný. Pro složité databázové aplikace z oblasti datových skladů je lepší platformou Oracle než MySQL. Na druhou stranu pro většinu „čtecích" webových aplikací nevyžadujících některé složitější funkcionality, může být lepší MySQL. Vždy záleží na konkrétních požadavcích, lepší tedy odpovídá pojmu kvalitnější ve smyslu vyhovění aktuálním potřebám a požadavkům. Nic na tom nemění ani informační bulletiny dodavatelů snažících se přesvědčit, že právě jejich platforma je tou ideální a nejlepší možnou.

[8] Grid je technologie, která umožňuje tzv. clusterování (spojování) více databázových serverů za účelem zvyšování výkonnosti a dostupnosti [15].
[9] Zkratka slov eXtensible MarkUp Language. The Extensible Markup Language (XML) je jednoduchý, velmi přizpůsobitelný textový formát odvozený od SGML (ISO 8879) [19].

Mezi přední dodavatele z pohledu podílu na trhu a využívání technologických novinek patří v oblasti univerzálně použitelných databázových technologií zejména společnosti Oracle, IBM, Microsoft, Sybase a Intersystems. První dvojici celkově patří téměř 70% databázového trhu. Některé průzkumy dosazují na první místo společnost Oracle, jiné společnost IBM [8].

Tabulka 1 - Zkoumané databázové systémy, zdroj: [vlastní].

Název	Aktuální verze	Výrobce	Internetové stránky
ORACLE	11g	Oracle	www.oracle.com
IBM DB2	9	IBM	www.ibm.com
MSSQL SERVER	2008	Microsoft	www.microsoft.com
SYBASE ADAPTIVE SERVER ENTERPRISE	9.0	Sybase	www.sybase.com
CACHÉ	2008.2	Intersystems	www.intersystems.com

3.3 DBS ORACLE

Databázové produkty společnosti Oracle patří mezi světovou špičku. Potvrzují to i nezávislé průzkumy přisuzující této firmě na trhu s databázovými technologiemi celkový podíl mezi 40 až 50 procenty. Vlajkový produkt této společnosti v oblasti databází představuje uvedená platforma Oracle11g [8].

Oracle databáze obsahuje řadu sofistikovaných technologií, unikátností, které již fungují od verze 10g. Nejvýznamnější z nich je podpora databázového zpracování ve gridu, založeného na využití výpočetního výkonu libovolného množství geograficky oddělených počítačů [8].

Zpracování ve gridu se již využívá u nedatabázových aplikací. V oblasti databázových technologií je ovšem tento přístup revoluční, a Oracle se tak konkurenci vzdaluje.

Se zpracováním ve gridu souvisí také podpora tvorby vysoce škálovatelných a nepřetržitě dostupných aplikací, pomocí technologie Real Application Cluster. Podstatná, při nasazení této technologie, je možnost škálování[10] i na levnějších strojích tvářících se jako clustery[11] a fakt, že v případě výpadku (plánovaného i náhodného) jednoho či více clusterů, není ovlivněna funkčnost provozované aplikace. V nejhorším případě může dojít k poklesu celkového výkonu – tento stav ale nemusí mnozí uživatelé ani zpozorovat [8].

Mezi další vlastnosti patří mnoho moderních databázových prvků. Od podpory objektových vlastností a XML, přes bezpečnost, zajišťovanou pomocí virtuálních privátních databází a analytické funkce pro datové sklady (dolování dat), až po podporu nestrukturovaných dat, prostřednictvím Internet File System a vysoce škálovatelných a nepřetržitě dostupných aplikací. Samozřejmostí je bezproblémová podpora národních prostředí, včetně správného třídění, dále podpora transakčního a distribuovaného zpracování či dotazovacího jazyka SQL [8].

Databázové platformy Oracle nejsou oblíbené jen díky vysokému výkonu a podpoře moderních technologií, ale také díky snaze maximálně usnadnit práci vývojářům a administrátorům. Tuto snahu již několik verzí potvrzuje, mimo jiné velmi mocné, administrátorské prostředí Oracle Enterprise Manager, pomocí kterého lze provádět komplexní správu celého prostředí (tedy několika serverů i databází, které navíc mohou být provozovány v geograficky odlišných lokacích), včetně plánování jednotlivých administrátorských činností a reakcí na problémové stavy [8].

Důležitá je integrace Oracle11g s ostatními produkty portfolia společnosti Oracle. Především jde o aplikační server a balík podnikových aplikací. V případě zájmu tak mohou uživatelé a vývojáři používat vyjma operačního systému vše od jedné společnosti. Zákazníci oceňují také maximálně dvou-procesorovou variantu Standard Edition One, která je nasměrována (i cenově), především do středně velkých podniků. Databázová platforma Oracle11g je k dispozici pro většinu dnes dostupných operačních systémů, Linux a Microsoft Windows nevyjímaje [8].

[10] Škálování, škálovatelnost slouží k měření, do jaké míry může počítač, služba nebo aplikace splňovat zvyšující se požadavky na výkon.
[11] Skupina spolupracujících počítačů.

3.4 MICROSOFT SQL SERVER

Společnost Microsoft již dávno není dodavatelem pouze operačních systémů či kancelářských aplikací, ale velmi dobře se jí daří také na poli vývojových nástrojů a databázových technologií. Vývojáři databázových aplikací již několik let berou platformu MS SQL Server 2005 jako zcela plnohodnotnou a jinak tomu zcela jistě není ani u verze MS SQL Server 2008, kterou Microsoft v září 2088 prezentoval na své Roadshow [8].

Mezi hlavní vlastnosti této platformy patří podpora vysokého výkonu, technologie XML nebo analytických principů používaných v rámci dolování dat. Díky podpoře internetových technologií a velkým možnostem v oblasti BI[12], jsou MSSQL Server 2005 i nově 2008 považováni za členy rodiny produktů .NET[13].

Vzhledem k tomu, že zejména s variantou Enterprise Edition se Microsoft zaměřuje na podnikovou sféru, nijak nepřekvapí ani podpora škálovatelnosti (MS SQL Server dokáže při symetrickém multiprocesingu[14] využít až 32 procesorů a 64 GB operační paměti), ani podpora tvorby nepřetržitě dostupných databázových aplikací [8].

Součástí distribuce MSSQL Serveru pochopitelně není pouze samotný databázový server, ale také řada dalších základních administrátorských nástrojů a klientů. Ve srovnání s mnohými výtvory konkurence je možné tyto produkty považovat za zdařilé, i když zdaleka ne za nejlepší. Obdobně jako samotný databázový server patří i základní aplikační okolí ke zlatému středu oceňovanému zejména administrátory starajícími se o středně složité databázové aplikace [8].

Především vývojáři menších aplikací, nevyžadujících konkurenční přístup mnoha uživatelů, využívají jádro MSDE (Microsoft Data Engine), které je uměle výkonově omezenou verzí databázového jádra popisovaného serveru – v MSDE je tedy podporována značná část funkčnosti platformy MS SQL Server. Novou variantou MSDE je SQL Server 2008 Express Edition.

---

[12] BI (Bussines Inteligence) – sada nástrojů, programů a řešení, pro podporu rozhodování [6].
[13] .NET je platforma pro běh nových distribuovaných aplikací [24].
[14] Symetrický multiprocesing: Každý procesor je v plánování samostatný. Má svou frontu připravených a sám rozhoduje, kterému procesu bude přidělen. Může nastat situace, kdy dva různé procesy na dvou různých procesorech chtějí modifikovat stejná data. Operační systém musí být v tomto případě naprogramován velmi opatrně. Je třeba zajistit, aby dva procesory nevybraly stejný proces. Zdroj: http://homen.vsb.cz/~kod31/vyuka/opsys/multilevel.html.

Určitým omezením se zdá orientace pouze na operační systémy od stejné firmy, která vadí především možným zákazníkům, kteří využívají unixové provozní platformy. Na druhou stranu představuje MSSQL Server produkt, který je vhodný pro všechny zájemce, kteří chtějí mít maximum produktů od jednoho dodavatele a snížit si tak riziko plynoucí ze vzájemné nekompatibility. Nechybí ani podpora systému Windows Server 2008 - Longhorn [8].

3.5 IBM DB2

Společnost IBM prezentuje na databázovém trhu produkt, který nese název DB2 Universal Database. Jde o produkt, který má za sebou více než dvě desítky let vývoje [8].

Produkt DB2 představuje relační databázovou platformu s kvalitní podporou standardů SQL, transakčního zpracování a bezpečnostních mechanismů, která ve variantě Enterprise Extended Edition plně vyhovuje požadavkům na vývoj a provoz vysoce výkonných, škálovatelných a nepřetržitě dostupných databázových aplikací. Vysoký výkon je přitom zajišťován především paralelním zpracováním (SMP – Symmetric Multi Processing i MPP – Massively Parallel Processors), nechybí ani možnost využití databázových clusterů, podpora dělení datových oblastí podle různých kritérií (partitioning) či automatického procesu výkonnostního ladění pomocí technologie SMART – Self Managing and Resource Tuning [11].

Součástí této platformy jsou také rozšíření vhodná pro zpracování multidimenzionálních či prostorových dat a umožňující nasazení databázové platformy v rámci analytických aplikací, např. při využití technologie dolování dat [8].

Aktuální verze DB2 umožňuje nejen zpracování klasických strukturovaných dat, ale také dat nestrukturovaných – od textu přes obrázky až po multimediální data. Vývojáři otevřených a rozlehlých databázových aplikací mají k dispozici pokročilé replikační technologie, navíc mohou při tvorbě webových aplikací využít podporu XML. Naprosto pochopitelná je také úzká spolupráce s moderními vývojovými nástroji, VisualAge for Java nevyjímaje [11].

Popisovaná databázová platforma je známa vedle kvalitativních vlastností také podporou řady provozních systémů. V případě operačních systémů jsou zastoupeny

takřka všechny významné produkty – od špičkových unixových systémů včetně HP-UX či IBM AIX a z/OS přes Windows včetně Windows Server 2008 [8].

3.6 ADAPTIVE SERVER

Společnost Sybase nabízí v rámci svého produktového portfolia řadu databázových produktů. Nosným článkem je robustní databázová platforma známá pod názvem Sybase Adaptive Server Enteprise (ASE). Sybase ASE, podporuje vedle běžně dostupných prvků (jazyk SQL, transakční zpracování, snadná administrace) také některé ne příliš časté vlastnosti. Mezi rozšiřující schopnosti lze zařadit například podporu zpracování transakcí v heterogenním databázovém prostředí či ladění příkazů v jazyce SQL, o které se stará SQL Debugger [8].

Jak již bývá pro databázové platformy této kategorie typické, nabízí i Sybase ASE podporu automatického přizpůsobování databázového serveru aktuálním požadavkům, mimo jiné Sybase ASE, využívá tzv. dynamickou optimalizaci výkonu. Podpora spočívá také v možnosti měnit výkonnostní nastavení databázového serveru (velikost vyrovnávací paměti, priority apod.) bez nutnosti zastavení serveru – tedy za plného provozu [8].

Velký důraz klade tato databázová platforma na problematiku zabezpečení. V Sybase ASE nechybí například omezení přístupu na úrovni konkrétního řádku, podpora zabezpečeného přenosu pomocí protokolu SSL, digitálních certifikátů podle standardu X.509 v3 či spolupráce se servery LDAP[15]. Pro zajištění zabezpečení na úrovni řádků využívá Sybase ASE technologii ACF (Application Context Facility) nabízející kontextově závislé klíče. Při přístupu k datům jsou navráceny pouze ty záznamy, které z bezpečnostního hlediska odpovídají klíči vygenerovanému při přihlášení uživatele. Stranou zájmu Sybase nezůstává ani podpora moderních databázových aplikací. Přínos Sybase ASE spočívá především v možnosti využít technologie XML, komponent Enterprise JavaBean a Javy přímo na úrovni databázového stroje. Poslední z uvedených předností umožňuje vývojářům tvorbu uložených procedur přímo v Javě, případně v kombinaci s jazykem SQL [8].

[15] LDAP (Lightweight Directory Access Protocol) je standardní protokol pro globální adresářové služby, fungující na principu klient-server [19].

Zájemci o databázovou platformu Sybase Adaptive Server Enterprise mají na výběr z mnoha provozních prostředí. Mezi podporované operační systémy patří především kvalitní unixy (HP-UX, IBM AIX apod.), stranou ale nezůstávají ani operační systémy společnosti Microsoft (serverová část ovšem vyžaduje řadu NT). Samozřejmostí je přitom podpora 64bitových variant, pokud tedy 64bitovou architekturu daný systém podporuje, a to i na Linuxu [8].

3.7 CACHÉ

Srdcem platformy Caché [kašé] je multidimenzionální architektura umožňující různé postrelační pohledy na spravovaná data. K datům lze díky této architektuře přistupovat nejen klasickým relačním způsobem, ale například také objektově, prakticky se všemi vlastnostmi známými z objektově orientovaných technologií. Fyzicky jsou data ukládána jako řídká vícerozměrná pole. Vedle objektového či relačního přístupu lze použít libovolnou další projekci zpřístupňující multidimenzionálně uložená data, například projekci do XML [9].

Z pohledu běžných vývojářů databázových aplikací se Caché tváří jako velmi pružné prostředí pro ukládání a správu dat, k dispozici je prakticky vše, co může být k vývoji a provozu databázové aplikace použitelné. Od transakčního zpracování a podpory konkurenčního přístupu k datům přes podporu dotazovacích jazyků a dotazovacích prvků v univerzálních programovacích jazycích až po podporu vícevrstvých aplikací či provozu 24^7^365. K dispozici jsou také nativní i univerzální databázová rozhraní a brány pro zajištění z hlediska vývojáře i uživatele transparentního přístupu k datům v konkurenčních platformách [9].

Platforma Caché nabízí také skriptovací technologii CSP (Caché Server Pages), která je svými možnostmi srovnatelná například s ASP či JSP[16]. Velkou výhodou oproti mnohé konkurenci je možnost využívat bitmapové indexy v rámci aplikací s klasickým transakčním zpracováním – tedy nejen pro OLAP[17], ale také pro OLTP[18] [9].

[16] Soubory na webovém serveru, které slouží k provozu interaktivních aplikací.
[17] Oblast databázových technologií podporujících rozhodování a analytické činnosti, obvykle nad datovými sklady.
[18] OLTP systémy uchovávají záznamy o jednotlivých uskutečněných transakcích a jsou obvykle realizovány pomocí dnes nejběžnější relační databázové technologie.

Součástí standardní distribuce je řada vyspělých nástrojů, základem je Caché Studio, které lze použít především k vývoji aplikací. Databázovým administrátorům je určen nástroj Caché Configuration Manager, pomocí kterého lze ovlivnit nejrůznější vlastnosti konkrétních instalací Caché. Databáze lze spravovat prostřednictvím nástroje Caché Control Panel. Pro práci s jazykem SQL slouží nástroj Caché SQL Manager, k dispozici je také terminálový nástroj Caché Terminal, který dovoluje řešit například velmi komplikované situace vyžadující přístup k databázím na co nejnižší úrovni [9].

Platforma Caché je k dispozici pro řadu operačních systémů, především pak pro přední Unixy a Windows. Vývojáři mají zdarma k dispozici jednouživatelskou verzi. Během posledních let se zlepšuje povědomí o Caché také v České republice, mezi největší řešení, co do počtu licencí, patří specializovaná aplikace v České spořitelně s více než 1 400 uživateli [9].

3.8 RESUMÉ

Databázové produkty jsou v jednom ohledu jako každé jiné. Existuje mnoho možností vzájemného srovnávání, ovšem jen s velkou nadsázkou lze tato srovnání považovat za objektivní.

Srovnávání databázových platforem je možné rozdělit do dvou hlavních oblastí:

- *technologické*
- *výkonnostní*

Z technologického hlediska je možné databázové produkty srovnávat především na základě nabízené funkčnosti, podpory progresivních technologií a souladu s průmyslovými standardy. Do úvahy při srovnávání je ale třeba brát nejen informační materiály výrobců, ale především praktické zkušenosti s danými produkty. Pokud není možné tyto zkušenosti získat z vlastní praxe či blízkého okolí, není zbytečné poohlédnout se po nějaké referenční instalaci či případové studii, které se občas objeví v odborném tisku.

Důležité je věnovat také pozornost míře podpory daných technologií. Všeobecně známým případem jsou klamavé informace o podpoře SQL 92. O tom, že produkt podporuje SQL 92 je možno se dočíst takřka v každém informačním materiálu,

zpravidla se ale jedná pouze o podporu na vstupní úrovni či určité podmnožiny tohoto standardu.

Z hlediska výkonu je možno vycházet především ze standardizovaných testů, existujte zde ovšem jedno velké *ale*. Nejpoužívanější výkonnostní testy, mezi které patří testy TPC[19], totiž jen málokdy odpovídají reálnému provozu (reálnému nasazení v praxi). Obvykle jsou tyto testy prováděny ve velmi omezeném časovém úseku a zjišťují tak pouze okamžitý výkon. K dispozici pak většinou nejsou informace o nárůstu zátěže, výpadcích, problémech se škálovatelností apod.

Jak již bylo řečeno, je srovnávání databázových platforem velmi problematickou a často neobjektivní záležitostí. Ona neobjektivnost plyne také z toho, že produkty tohoto charakteru by neměly být srovnávány samy o sobě, ale vždy na základě toho, k čemu má dané řešení sloužit. Jinými slovy pro jednoduchou aplikaci může být kvalitnější MS FoxPro než Oracle. Pod kvalitou se totiž skrývá především vhodnost daného produktu na splnění požadavků kladených na řešení.

Pokud je zapotřebí z jakéhokoli důvodu databázové produkty srovnávat, je nutné dodržet následující fakta:

- srovnávání by mělo vycházet z konkrétních požadavků,
- srovnávání je nutno brát jako subjektivně,
- není možné věřit pouze informacím poskytnutým výrobcem.

V roce 2008 pořádal informační portál o databázových technologiích Databázový svět[20] ve spolupráci s časopisem Computing rozsáhlý průzkum využití databázových technologií v České republice a na Slovensku. Průzkum Databáze 2008 byl rozdělen mezi odbornou komisi a čtenářskou veřejnost. [11].

Ocenění odborné komise získal pro rok 2008 produkt Oracle Database 11g Enterprise Edition společnosti Oracle. Odborná komise byla složena ze 14 zástupců komerční a akademické sféry:

- Pavel Barták, EMC Czech Republic,

[19] Total product cost – jsou to celkové náklady na daný produkt.
[20] http://www.dbsvet.cz.

- Prof. Ing. Ladislav Buřita, CSc., Univerzita obrany, Fakulta vojenských technologií,
- Prof. Dr. Jiří Hřebíček, Institut biostatistiky a analýz,
- Ing. Zbyněk Jankovský, Cleverlance Enterprise Solutions,
- Ing. Ladislav Juřenčák, Per4mance,
- Ing. Ján Letko, Forza,
- Tomáš Mlčoch, Kvintech,
- RNDr. Petra Poulová, Univerzita Hradec Králové, Fakulta informatiky a managementu,
- Doc. Ing. Karel Richta, CSc., Univerzita Karlova v Praze, Matematicko-fyzikální fakulta,
- Kamil Řeháček, Amenit,
- Ing. Bc. Jaroslav Šmarda, MBA, Vema,
- Ing. Michal Trunda, PhD., Ministerstvo obrany,
- Petr Zahradník, Computer Laboratory,
- Doc. Ing. Jaroslav Zendulka, CSc., Vysoké učení technické v Brně, Fakulta informačních technologií.[11]

V rámci čtenářské části soutěže bylo odevzdáno přes 2 000 platných hlasů a ocenění získaly tyto produkty (s určením pořadí):

1. Caché 2008.1 společnosti InterSystems
2. IBM Informix Dynamic Server 11.5 Enterprise Edition společnosti IBM
3. MS SQL Server 2008 Enterprise Edition společnosti Microsoft

Mimořádné ocenění redakce Databázového světa bylo uděleno produktu IBM Informix Dynamic Server 11.5 Enterprise Edition společnosti IBM [11].

Zajímavý je také výsledek řezu zaměřeného na trojici nejpoužívanějších platforem (dle licencí) v daném podniku, do úvahy byly v tomto případě brány všechny

existující verze dané platformy. Nejčastěji byla do této trojice zařazena platforma Oracle Database a další dvě příčky patří platformám IBM DB2 a Informix DS [20].

Komerčním platformám nicméně pravděpodobně příliš nesvědčí úroveň technické podpory, neboť uživatelé open source platforem jsou s dostupnou technickou podporou více než čtyřnásobně spokojenější. Zajímavostí pak může být informace, že na Moravě je poměr využití komerčních platforem versus open source platforem ve srovnání s českými kraji nižší takřka o 40 % [11].

Podíly jednotlivých dodavatelů databázových technologií se mění jak podle oblasti využití databázových platforem, tak i podle použitých technologií. Zaměříme-li se pouze na produkty plnohodnotně využívající objektové technologie, bude největší podíl na trhu patřit databázové platformě Caché od společnosti Intersystems.

V oblasti nativních platforem v Javě patří přední příčka produktu JDataStore společnosti Borland. Mezi nativními XML platformami zase patří přední příčka produktu Tamino společnosti Software AG. Budeme-li brát do úvahy platformy a nástroje zaměřené výhradně na oblast analytického zpracování, dostane se do popředí například společnost SAS.

Databázový trh je tedy z pohledu žebříčků velmi podobný trhům ostatním – od potravinového přes hodinářský až po automobilový. Nikdy ale nesmí být opomenuta skutečnost, že důležitá je především míra splnění požadavků kladených na databázovou platformu konkrétních aplikací [20].

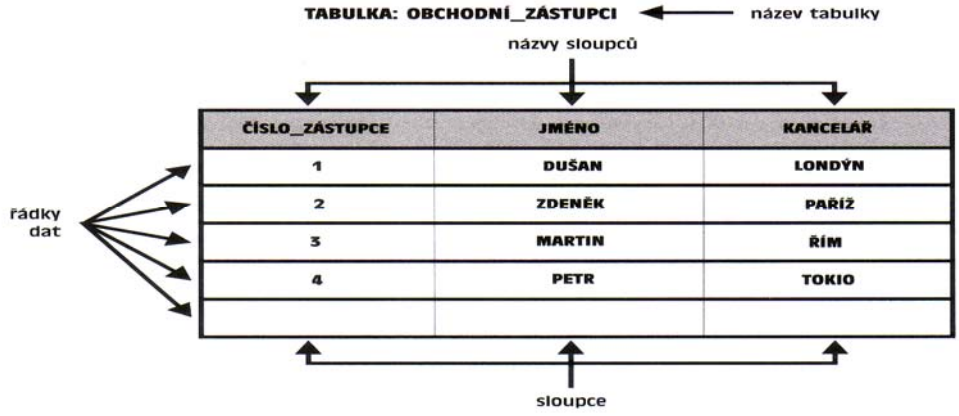
4 DBS ORACLE A JEHO VLASTNOSTI

Jelikož Magistrát města Pardubic požaduje podle interních dokumentů informační a bezpečnostní politiky systém, který je stabilní, maximálně dostupný a v největším ohledu bezpečný, volba databázového prostředí Oracle byla tou správnou variantou. A jelikož dodavatelské firmy nabídly své řešení na platformě Oracle, stal se databázový systém Oracle provozní databázovou platformou informačního systému. Vyvstává otázka, proč je systém Oracle vyhovujícím řešením? V následující kapitole je popsána struktura a vlastnosti, které tvrzení o bezpečnosti, stabilitě, škálovatelnosti a vysoké dostupnosti potvrzují [14].

4.1 POPIS

Databáze je sada dat. Oracle umožňuje ukládat data a získávat přístup k nim způsobem odpovídajícím definovanému modelu, který je znám jako relační model dat. Z tohoto důvodu je Oracle považován za systém správy relační databáze (za relační systém řízení báze dat). Většina odkazů na databázi se vztahuje nejen k fyzickým datům, ale také ke kombinaci fyzických a paměťových objektů a objektů procesů [14].

Data jsou v databázi ukládána do tabulek. Relační tabulky lze definovat sloupci s příslušným názvem. Data jsou ukládána jako řádky tabulky. Mezi tabulkami je možné vytvořit vzájemné vazby [14]. Viz. obrázek 3.



Obrázek 3 - Příklad struktury tabulky, zdroj: [14].

Kromě uložení dat v relačním formátu podporuje Oracle objektově orientované struktury, například *abstraktní datové typy a metody*. Objekty lze provázat s jinými objekty nebo také mohou obsahovat další objekty. Objektové pohledy mohou být použity k povolení objektově orientovaných rozhraní pro data, aniž by se v tabulkách musely vytvářet jakékoli změny [14].

Bez ohledu na to, zda se využije relační nebo objektově orientovaná struktura, ukládá server Oracle data do souborů. Data jsou interními strukturami databáze logicky mapována na soubory. Různé typy dat lze tedy ukládat samostatně. Tato logická rozdělení se nazývají tabulkové prostory (Tablespace) [14].

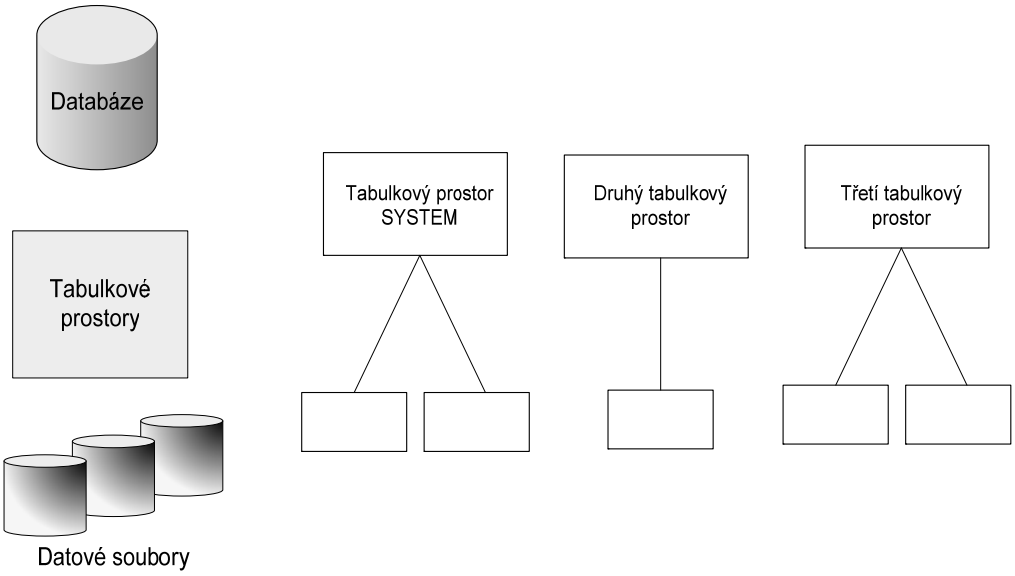
4.2 TABULKOVÉ PROSTORY - TABLESPACE

Tabulkový prostor tedy představuje logické rozdělení databáze. Všechny databáze obsahují alespoň jeden tabulkový prostor (SYSTEM). Seskupením uživatelů nebo aplikací pomocí dalších tabulkových prostorů lze usnadnit údržbu a výkon databáze. Tabulkový prostor může být přiřazen pouze k jedné databázi [14].

4.3 DATOVÉ SOUBORY

Jednotlivé tabulkové prostory se skládají z jednoho nebo více souborů na disku, tzv. *datových souborů*. Datový soubor může patřit pouze k jednomu tabulkovému prostoru. Po vytvoření tabulkových prostorů lze měnit jejich velikost. Vytvoření nových tabulkových prostorů vyžaduje vytvoření nových datových souborů. Po přidání datového souboru do tabulkového prostoru není možné tento datový soubor z tabulkového prostoru odstranit a nelze jej přiřadit k žádnému jinému tabulkovému prostoru [14].

Pokud se uloží objekty databáze do více tabulkový prostorů, mohou se fyzicky oddělit umístěním příslušných datových souborů na samostatné disky. Oddělení dat představuje důležitý nástroj při plánování a optimalizaci zpracování vstupních a výstupních požadavků na databázi [14]. Vztahy mezi databázemi, tabulkovými prostory a datovými soubory jsou znázorněny na obrázku 4.



Obrázek 4 - Vztah mezi databázemi, tabulkovými prostory a datovými soubory, zdroj: [vlastní]

Jak již bylo uvedeno, datové soubory poskytují fyzické úložiště pro data databáze. Datové soubory lze tedy považovat za interní struktury, protože jsou přímo závislé na tabulkových prostorech a zároveň jako externí struktury, protože představují fyzické soubory [14].

Od datových souborů jsou odděleny následující typy souborů, přestože mají vazby s databází [14].

- Transakční protokoly (redo logs).
- Řídící soubory (control files).
- Trasovací soubory a protokoly upozornění (alert log).

4.4 TRANSAKČNÍ ŽURNÁL

Oracle uchovává protokoly všech *transakcí* prováděných vůči databázi. Tyto transakce jsou zaznamenávány do souborů, které se nazývají *online soubory transakčního protokolu (aktivní soubory transakčního protokolu, transakčního žurnálu).* Soubory protokolu se v případě selhání databáze používají k obnovení databáze pomocí informací o provedených transakcích. Informace o souborech transakčního protokolu jsou ukládány mimo datové soubory databáze [14].

Soubory protokolu také umožňují systému Oracle zefektivnit způsob zápisu dat na disk. Transakce, která se v databázi objeví, je vložena do mezipaměti protokolu. Datové bloky ovlivněné touto transakcí tedy nejsou okamžitě ukládány na disk [14].

Všechny databáze Oracle obsahují tři nebo více aktivních souborů transakčního protokolu. Oracle zapisuje do těchto souborů cyklickým způsobem. Po zaplnění prvního se začne automaticky plnit druhý, než je opět zaplněn. Po zaplnění všech souborů protokolu se Oracle vrátí zpět k prvnímu souboru a začne přepisovat jeho obsah novými daty o provedených transakcích. Databáze spuštěná v režimu *ARCHIVELOG* vytváří kopie aktivních souborů ještě před tím, než je přepíše. Tyto archivní soubory lze potom využít k obnovení libovolné části a stavu databáze [14].

Soubory transakčního protokolu lze zrcadlit (replikovat). Zrcadlení těchto aktivních souborů je nezávislé na operačním systému nebo na hardwarových možnostech operačního prostředí [14].

4.5 ŘÍDÍCÍ KONTROLNÍ SOUBORY

Celková fyzická architektura databáze je uchovávaná v řídících souborech (control files). Tyto soubory zaznamenávají informace o řízení všech souborů v databázi. Řídící soubory udržují interní konzistenci a řídí operace vedoucí k zotavení databáze v případě jejího selhání. Pokud nastane ztráta řídícího souboru, zpravidla to znamená ztrátu celé databáze. Existují sice metody zotavení bez řídících souborů, ale úspěšnost se pohybuje na hranici 30%. Kopií řídících souborů se ukládá více, protože jak již bylo řečeno, jsou pro databázi velmi důležité. Tyto soubory jsou obvykle uchovávány na samostatných discích, aby nedošlo k jejich ztrátě při selhání disků. Databáze vytváří a uchovává řídící soubory specifikované při založení databáze.

Názvy řídících souborů databáze jsou určovány podle inicializačního parametru CONTROL_FILES, který se nachází v iniciačním souboru *INIT.ORA* [21][14].

[21] Soubor INIT.ORA je inicializační soubor databáze Oracle, který obsahuje seznam argumentů, jenž mají být při spuštění databáze uskutečněny.

4.6 TRASOVACÍ SOUBORY A SOUBORY UPOZORNĚNÍ

Ke všem procesům na pozadí spuštěným v instanci, je přidružen trasovací soubor. Trasovací soubor obsahuje informace o důležitých událostech probíhajících na pozadí. Kromě těchto trasovacích souborů Oracle uchovává soubor, který se nazývá protokol upozornění (alert log). Do tohoto souboru jsou zaznamenávány příkazy a výsledky příkazů hlavních událostí týkajících se fungování databáze. Je možné v něm například najít údaje o vytváření tabulkových prostorů, přepínání souboru transakčního žurnálu, operacích obnovy nebo spouštění databáze. Soubor upozornění je podstatný zdroj informací pro každodenní správu databáze. Trasovací soubory se nejvíce ocení při zjišťování příčin kritického selhání databáze [14].

Protokol upozornění je nutné sledovat každý den. Položky v něm mohou informovat o všech problémech, které se objeví během operací s databází, včetně všech interních chyb.[22] Práci s protokolem upozornění si lze usnadnit, pokud se povolí jeho každodenní automatické přejmenování. Jestliže je například název protokolu upozornění alert_orcl.log, lze ho přejmenovat tak, aby název souboru obsahovat aktuální datum. Při dalším pokusu o zápis do protokolu upozornění Oracle nenajde soubor s názvem alert_orcl.log a databáze vytvoří nový soubor. Potom budete mít k dispozici aktuální i předchozí protokol upozornění. Pokud se takto rozdělí položky protokolu upozornění, bude to vést k lepší možnosti analýzy [14].

4.7 INSTANCE

K získání přístupu k datům v databázi používá Oracle několik procesů na pozadí, které sdílejí všichni uživatelé. Kromě toho existují paměťové struktury (souhrnně označované jako SGA[23]), které uchovávají poslední data dotazovaná z databáze. Největší sekce SGA, vyrovnávací paměť, Shared SQL Pool, Large Pool a Java Pool obvykle zabírají 95% paměti přidělené SGA. Tyto oblasti paměti umožňují vylepšit výkon databáze, protože zmenšují počet vstupně – výstupních procesů prováděných vůči datovým souborům [14].

[22] Příklad chyby - ORA-0600.
[23] System Global Area – paměťové struktury v databázi Oracle..

Instance databáze (neboli server) je sada paměťových struktur a procesů na pozadí, která má přístup k databázovým souborům. K jedné databázi může mít přístup více instancí[24]. Vztah mezi instancemi a databázemi je znázorněn na obrázku č. 5.

Obrázek 5 - Instance a datové soubory v systému Oracle, zdroj: [vlastní]

Parametry určující velikost a složení instance jsou ukládány do inicializačního souboru s názvem init.ora nebo jsou uchovávány v databázi v souboru s parametry serveru, který je označován SPFILE a je uložen v souboru spfile.ora.

Při spuštění instance je načten soubor s inicializačními parametry, který může být změněn databázovým administrátorem. Libovolné úpravy provedené v inicializačním souboru se projeví až při dalším spuštění instance. Název inicializačního souboru instance obvykle obsahuje název instance. Jestliže je název instance ORCL, bude se soubor obvykle nazývat initOrcl.ora. V databázi Oracle je stále více parametrů, které lze nastavit dynamicky. Bez ohledu na obsah souboru s inicializačními parametry mohou tyto změny parametrů zůstat aktuální i po restartu databáze. U Oracle9i může databázový administrátor místo souboru init.ora použít

[24] RAC – Real application cluster – Technologie společnosti Oracle uvedená ve verzi Oracle9i umožňující snadné využívání clusterů a zajišťující v případě výpadku jednoho či několika uzlů stoprocentní dostupnost celé databáze (tedy nejen části), důsledkem je pouze pokles výkonu.

soubor SPFILE a zajistit, aby byly změny inicializačních parametrů dočasné nebo trvalé [14].

5 VYUŽITÍ DBS ORACLE V INSTITUCI VEŘEJNÉ SPRÁVY

5.1 CHARAKTERISTIKA INSTITUCE

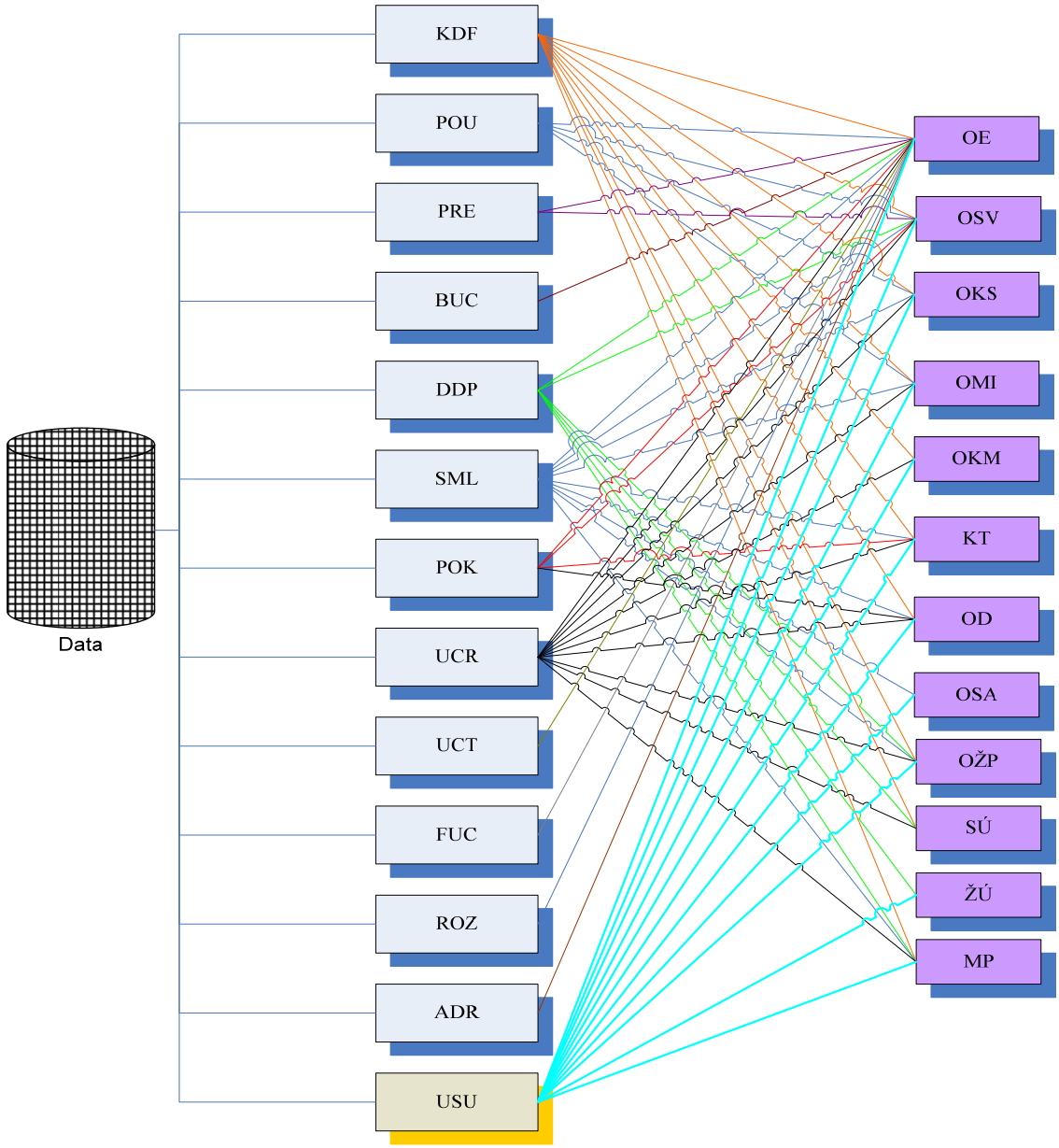
Název	Magistrát města Pardubic
Sídlo	Pernštýnské náměstí 1 530 21 Pardubice
Identifikace	00274046
Právní forma	Veřejná správa a samospráva
Internetové stránky	http://www.pardubice.eu
Primátor	Ing. Jaroslav Deml
Tajemník	Mgr. Martin Růžička

5.2 PROFIL IS MAGISTRÁTU MĚSTA PARDUBIC

Informační systém magistrátu je složen z několika dílčích, robustních informačních systémů, které navzájem díky integraci tvoří celek – Informační systém Magistrátu města Pardubic.

První část celého systému je systém Ginis od společnosti Gordic, spol. s r.o. Tento systém zahrnuje veškeré ekonomické agendy, jako jsou daně, dávky, pohledávky (DDP), knihu došlých faktur (KDF), rozpočtování (UCR), pokladní systém (POK), účetnictví (UCT), evidenci smluv (SML) a dále spisovou agendu, jejíž součástí je program na vedení spisové služby (USU), podatelna (POD), výpravna (VYP). Celý systém Ginis je stavěn na architektuře klient-server a jako celek je provozován nad databázovým systémem Oracle.

Pro přehled agend je přiloženo schéma informačního systému Ginis, tak, jak je implementován přímo v instituci Magistrátu města Pardubic a jak byl specialisty IT navrhnut.

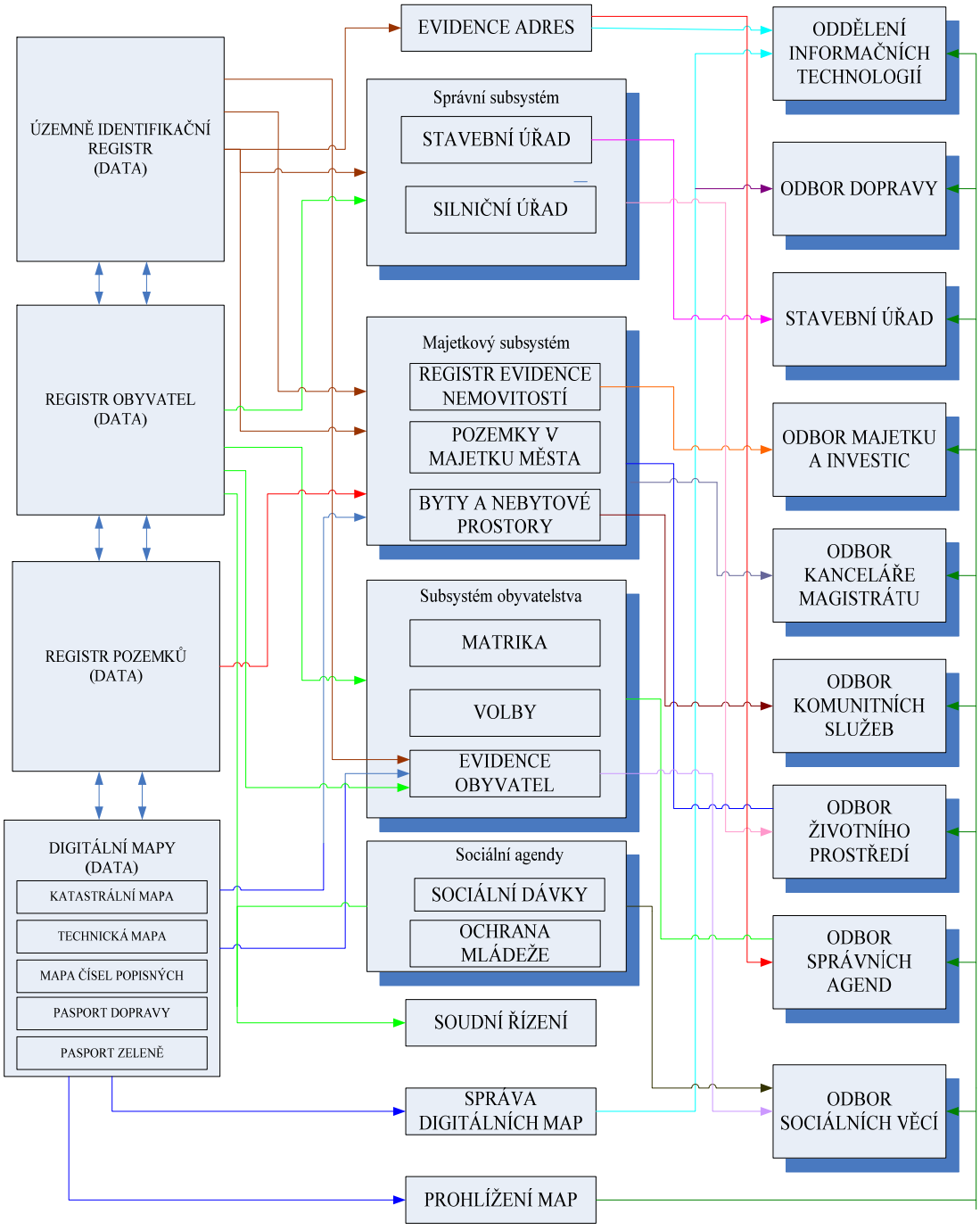


Obrázek 6 - Schéma systému Ginis, zdroj: [vlastní].

Druhou stěžejní částí informačního systému jsou aplikace od společnosti Geovap, spol. s r.o., které jsou souhrnně označované jako Cityware. Jedná se především o registr obyvatel (ROB), územně identifikační registr (UIR), geografický informační systém (GIS) a sociální agendu (SOC). Integrací systémů bylo dosaženo například provázanosti aplikace Smlouvy (SML) a registru obyvatel (ROB). Celý systém registrů

je opět postaven na architektuře klient-server a je provozován nad databázovým systémem Oracle.

Jako poslední prvek celého systému, který je rovněž vyvíjen nad databází Oracle, je sada aplikací dalších firem, jako je Vita (Stavební úřad a aplikace Přestupky) nebo Canon (Windream – elektronický archiv stavebního úřadu). Snahou magistrátu je neustále integrovat jednotlivé aplikace mezi sebou a vytvářet tak ucelený a komplexní systém. V současné době se připravuje integrace programu Vita – stavební úřad s aplikací Univerzální spisový uzel (USU) od firmy Gordic nebo provázání elektronické podatelny, jejímž dodavatelem je společnost PVT, a.s. rovněž s Universálním spisovým uzlem.



Obrázek 7 - Schéma informačního systému Cityware, zdroj: [vlastní]

5.3 ZÁLOHOVÁNÍ DAT A JEJICH OBNOVA

Zálohování je stěžejním místem databázového systému. Podle toho, jak se databáze bude zálohovat ať již svépomocí nebo s pomocí operačního systému, taková bude spolehlivost zachování dat a zaručená kontinuita jejich zpracování bez větších

výpadků. Každá společnost považuje za nejcennější své know-how a své podnikatelské záměry. Stejně důležitá jsou ovšem data a v nich obsažené informace, která každá společnost produkuje v ekonomických systémech, v rámci elektronické komunikace, v kancelářských aplikacích, atd. Důležitosti dat by měla odpovídat úroveň zálohování. Ne vždy tomu však tak skutečně je.

Nejdůležitější v celém procesu zálohování je správné určení dat, která se mají zálohovat. Samozřejmostí by měla být záloha všech dat, která jsou v rámci dané organizace zpracovávána. Jakékoli omezení zálohování na první pohled nedůležitých dat se může v budoucnu značně nevyplatit. Ze zálohovacího procesu lze některá data vyřadit, ovšem mělo by být vždy stoprocentně jasné, proč není nutné tato data zálohovat.

Otázka nezní pouze jaká data zálohovat, ale je důležité určit, i kdy se data budou zálohovat. Ačkoli je jasné, že se jedná o zásadní otázky, firmy si plně neuvědomují, co tyto otázky znamenají. Když už se firma rozhodne zálohovat, dopouští se v řadě případů fatálních chyb, které ve finální fázi mohou mít stejný výsledek, jako kdyby vůbec nezálohovaly.

Určité „ukolébání“ může fungovat v případech, kdy nebylo nutné zálohy, v horizontu několika let, použít.

S vývojem informačních technologií, se vyvíjejí a mění i pravidla pro zálohování. V případech, kdy nastane na první pohled nevýznamná změna v informační architektuře podniku, je nutné celý proces zálohování přehodnotit a zkontrolovat, zda zálohovaná data odpovídají datům, která chce podnik zachovat. Typickým případem je nasazení nového serveru. Po úspěšné implementaci musí následovat zařazení nového systému do procesů zálohování.

V podniku by měla být sepsán dokument Bezpečnostní politika, který musí mimo jiné obsahovat i pravidla zálohování. Musí být komplexní. Na druhou stranu ale musí umožnit snadnou změnu. Samozřejmostí jsou i jasně vymezené kompetence i povinnosti pro jednotlivé konkrétní osoby. Přijatá pravidla je také velmi vhodné průběžně kontrolovat. Dokument Bezpečností politika by měl rovněž předepisovat i pravidelné, cyklické zkoušky obnovy ze záloh, aby bylo jisté, že v případě výpadku jsou zálohy funkční.

Jednotlivé zálohy by měly být uloženy na místě geograficky odlišném od místa, na kterém se nachází zálohovaná data. V opačném případě je riziko současného poškození nebo odcizení dat i jejich záloh velmi vysoké. Požár, povodeň nebo orkán jsou dost pádné důvody.

Požadavek geograficky odlišných lokalit lze zabezpečit různými způsoby. V malých firmách je velmi obvyklé, že si vedoucí nebo jiný pracovník nosí pravidelně zálohy domů, větší společnosti se spoléhají na speciální či bankovní úložny. Geografická vzdálenost by měla být v dost velké vzdálenosti od firmy, aby nemohla být zasažena nenadálou událostí typu požár, ale není rovněž vhodné, aby byla příliš vzdálená, což může působit problémy při obnově. Studie ukazují, že hodina výpadku může ve velkých společnostech znamenat i sta tisícové, mnohdy i miliónové ztráty.

V reálném provozu se ukazuje jako výhodné pořizovat zálohy duplicitně. Jedna kopie zůstává v místě pořízení a druhá kopie je převezena do bezpečné, vzdálené lokality. Většina zálohovacích systémů umožňuje již při procesu zálohování určit, kolik kopií má být vytvořeno.

Samozřejmostí je, že výše zmíněná pravidla musí platit, ať si firma zajišťuje zálohování svépomocí nebo se firmě o zálohování stará externí, specializovaná firma. Možnost využití externích specializovaných firem pro zálohování, bývá zpravidla výhodné pro malé firmy, které spravují velká množství dat. Velké společnosti, mezi které lze započítat i Magistrát města Pardubic, si zálohování řeší svépomocí. Většinou disponují s velmi silným IT zázemím. Ale i v těchto společnostech se vhodné konzultovat zálohovací postupy s externími firmami.

Tato část práce je již konkrétně zaměřena na databázový systém Oracle a jeho implementaci v rámci Magistrátu města Pardubic. Platforma Oracle zajišťuje celou řadu zálohovacích procedur a možností, které pomáhají zabezpečit databázi. Pokud jsou správně implementovány, pomohou tyto možnosti efektivně zálohovat databáze a snadno a efektivně je obnovovat ze zálohy. Mezi zálohovací možnosti databáze Oracle patří logické a fyzické zálohy, přičemž pro obě z nich je k dispozici řada možností. Tato kapitola se nebude dopodrobna zabývat všemi možnostmi a všemi scénáři z hlediska obnovování ze záloh. Všechny tyto informace lze získat v dokumentaci k platformě Oracle. Tato kapitola je primárně zaměřena na využití nejvhodnější možnosti tím

nejefektivnějším možným způsobem. Je zde uvedeno, jak lze nejlépe integrovat dostupné zálohovací procedury mezi sebou a se zálohami v rámci operačního systému.

5.4 MOŽNOSTI A ZPŮSOBY ZÁLOHOVÁNÍ

Pro zálohování databáze Oracle se používají tři standardní metody: exporty, zálohy offline a zálohy online. Export je *logická* záloha databáze. Zbývající dvě metody zálohování jsou *fyzické* zálohy souborů. Následující části jsou zaměřeny na úplný popis těchto možností. Fyzické zálohy je možné provádět pomocí uživatelských skriptů nebo pomocí nástroje RMAN (Oracle Recovery Manager) [14].

Tabulka 2 - Typy záloh, zdroj: [14].

Metoda	Typ	Charakteristické vlastnosti obnovení
Export	Logická	Pokrývá všechny databázové objekty do momentu exportu.
Zálohy offline	Fyzická	Pokrývá databázi do okamžiku jejího vypnutí. Pokud byla databáze spuštěna v režimu ARCHIVELOG, je možné databázi obnovit ze zálohy do původního stavu v kterémkoli okamžiku.
Zálohy online	Fyzická	Pokrývá databázi v libovolném okamžiku.

Robustní zálohovací strategie zahrnuje fyzické i logické zálohy. Obecně je možné říci, že provozní databáze používá jako primární zálohovací metodu fyzické zálohy a logické zálohy se používají jako sekundární metoda. Pro vývojářské databáze a pro některá zpracování malých objemů dat je možné provádět logické zálohy. Je třeba pochopit důsledky a použití fyzických i logických záloh, aby bylo možné pracovat na vývoji nejvhodnějšího řešení pro databáze. V následující tabulce je uveden návrh integrace všech typů záloh v závislosti na používané databázi [14].

Tabulka 3 - Integrace logických a fyzických záloh, zdroj: [14].

Typ databáze	Zálohy online	Zálohy offline	Export
Libovolná velikost, velké množství transakcí.	V noci	Jednou za týden	Jednou za týden
Malá databáze, většinou jen pro čtení.	Neprovádí se	V noci	V noci
Velká databáze, většinou jen pro čtení	Neprovádí se	V noci	Jednou za týden

Logická záloha databáze zahrnuje čtení sady databázových záznamů a jejich zápis do souboru. Tyto soubory jsou čteny nezávisle na svém fyzickém umístění. V databázi Oracle provádí tento typ záloh databází nástroj pro export. K obnovení pomocí souboru generovaného exportem se používá nástroj pro import.

Z hlediska zálohování respektive obnovy databáze Oracle rozlišujeme několik způsobů zálohování resp. obnovy databázových dat [14].

5.5 VARIANTY ZÁLOHOVÁNÍ A OBNOVY

5.5.1 Export, import

Nástroj pro export databáze Oracle zadává dotazy na databázi, včetně systémového katalogu, a zapisuje výstup do *binárního výstupního souboru exportu.* Může exportovat celou databázi, konkrétní uživatele nebo konkrétní tabulky. Během exportu je možné zvolit, zda mají být exportovány informace systémového katalogu přidružené k tabulkám (například udělená oprávnění, indexy a omezení). Soubor zapsaný programem pro export bude obsahovat příkazy nezbytné k opětovnému vytvoření všech zvolených objektů a dat [14].

Od verze Oracle9i je možné provádět export na úrovni tabulkových prostorů a exportovat tak všechny objekty, které daný tabulkový prostor obsahuje. Všechny indexy definované k exportovaným tabulkám jsou také exportovány. U exportů na úrovni prostorů používají klauzuli **tablespace** nástroje pro export, jak je popsáno dále v této kapitole. Po exportování je možné data importovat prostřednictvím nástroje pro import. Nástroj pro import čte binární výstupní soubor exportu vytvořený nástrojem pro export a provede příkazy, které zde nalezne. Může jít například o příkaz **create table** a následně příkaz **insert** k naplnění dat do tabulky [14].

Data, která byla exportována, není třeba importovat do stejné databáze ani do stejného schématu, které se používalo ke generování výstupního souboru exportu. Pomocí výstupního souboru exportu je možné vytvořit duplicitní sadu exportovaných objektů v jiném schématu nebo v jiné databázi [14].

Je možno importovat veškerá exportovaná data nebo pouze jejich část. Pokud se bude importovat celý výstupní soubor exportu, který je výsledkem úplného exportu, budou během importu vytvořeny všechny databázové objekty (včetně tabulkových prostorů, datových souborů a uživatelů). Často je však vhodnější předem vytvořit příslušné tabulkové prostory, uživatele a určit tak fyzické rozdělení objektů v databázi [14].

Pokud se bude importovat pouze část dat z výstupního souboru exportu, je třeba před importem nastavit tabulkové prostory, datové soubory a uživatele, kteří budou vlastnit a ukládat data. Exportovaná data je možné importovat do databáze Oracle další vyšší verze. Obecně to lze provést i zpětně, tj. import z exportního souboru vytvořeného v novější hlavní verzi, bude však třeba provést další činnosti, aby byly podporovány starší verze pohledů, které nástroj pro export používá. Podrobné informace o požadavcích na používání programu pro export či import v různých verzích jsou uvedeny v souborech README.doc [14].

5.5.1.1 Exportovací nástroj

Nástroj exportu má čtyři úrovně funkčnosti: úplný režim, režim tabulkového prostoru, uživatelský režim a režim tabulky. Oddíly je možné exportovat prostřednictvím modifikované verze exportů v režimu tabulky.

V úplném režimu se exportuje celá databáze. Je přečten celý systémový katalog a příkazy DDL[25], potřebné pro opětovné vytvoření úplné databáze, jsou zapsány do výstupního souboru.

Tento soubor obsahuje příkazy vytváření pro všechny tabulkové prostory, všechny uživatele a všechny objekty, data a oprávnění v jejich schématech.

V režimu tabulkového prostoru budou exportovány všechny objekty obsažené v zadaném tabulkovém prostoru, včetně definice indexů k objektům, které jsou obsaženy, i to, zda jsou v jiném tabulkovém prostoru.

V uživatelském režimu se exportují objekty uživatele a také data, která tyto objekty obsahují. Budou se také exportovat všechna oprávnění a indexy vytvořené uživatelem k objektům uživatele. Oprávnění a indexy vytvořené jinými uživateli, než je vlastník objektu, se neexportují.

V režimu tabulky je exportována zadaná tabulka. Struktura, indexy a oprávnění k tabulce jsou exportovány společně s daty i bez nich. V režimu tabulky je také možné exportovat úplnou sadu tabulek vlastněných uživatelem (zadáním vlastníka schématu bez názvů tabulek). Lze také zadat oddíly tabulky, které je žádoucí exportovat.

Export je možné spustit interaktivně prostřednictvím nástroje OEM nebo prostřednictvím souborů s příkazy.

Řada parametrů je mezi sebou v konfliktu nebo tyto parametry mohou vést k nekonzistentním instrukcím pro program export. Například nastavení hodnoty **full=y** a **owner=hr** by selhalo, protože parametr **full** volá export v úplném režimu, zatímco parametr **owner** určuje, že má být proveden export v uživatelském režimu.

Parametry programu *export* je možné zobrazit online prostřednictvím následujícího příkazu příkazové řádky:

exp help=y

Volba **compress=y** modifikuje parametr **initial** klauzule **storage** pro segmenty, které mají více rozsahů. Při následném importu bude celkové přidělené místo komprimováno do jediného rozsahu. K této funkčnosti je třeba uvést dva důležité body:

[25] Data definition language, jazyk pro definování datových struktur.

- Za prvé platí, že se komprimuje *alokované*, nikoliv využité místo. Prázdná tabulka s 300 MB alokovanými ve třech 100 MB rozsazích bude komprimována do jediného 300 MB rozsahu. Nezíská se tak žádné místo.
- Za druhé platí, že pokud tabulkový prostor obsahuje několik datových souborů, může segment alokovat místo, které je větší, než velikost datového souboru. V takovém případě by použití hodnoty **compress=y** způsobilo změnu klauzule **storage** s velikostí rozsahu **initial** větší, než je velikost jakéhokoli datového souboru. Jelikož jeden rozsah nemůže používat více než jeden datový soubor, dojde při importu k selhání při vytváření objektů.

V následujícím příkladě je použita volba **compress=y** při exportování vlastníka HR a THUMPER [14].

exp system/heslo file=expdat.dmp compress=y owner=(HR, THUMPER)

5.5.1.2 Importovací nástroj

Nástroj pro import čte výstupní soubor exportu a spustí příkazy, které jsou v něm uložené. Pomocí importu je možné selektivně obnovit ze zálohy objekty nebo uživatele z výstupního souboru exportu.

Import je možné spustit interaktivně nebo přes příkazové soubory. Řada parametrů importu je mezi sebou v konfliktu, v jiném případě mohou vést k nekonzistentním instrukcím pro program importu. Například nastavení hodnoty **full=y** a **fromuser=hr** by selhalo, protože parametr **full** volá import v úplném režimu, zatímco parametr **fromuser** určuje, že má být proveden import v uživatelském režimu. Parametr **destroy** je velmi užitečný pro databázové administrátory, kteří na jednom serveru provozují několik databází. Jelikož úplné exporty databází zaznamenávají celý systémový katalog, zapisují se definice tabulkových prostorů a datových souborů do výstupního souboru exportu. Definice datových souborů budou zahrnovat úplné cesty k souborům. Pokud se bude tento výstupní soubor exportu používat pro přenos dat do zvláštní databáze na stejném serveru, může dojít k problémům [14].

Problém je v tom, že při importu do druhé databáze z úplného exportu první databáze provede program importu příkazy **create tablespace** uvedené ve výstupním souboru exportu. Tyto příkazy budou nést pro databázi instrukce o tom, aby se vytvářely soubory ve stejném adresáři, se stejnými názvy jako soubory z první databáze. Bez použití parametru **destroy=n** by mohlo dojít k přepsání datových souborů první databáze [14].

5.5.1.3 Návratové segmenty (Undo)

Ve výchozím nastavení vydá server příkaz **commit** po každém dokončeném importu tabulky. Výsledný návratový segment bude obsahovat **RowID** pro všechny importované řádky. Chceme-li snížit počet návratových segmentů, zadáme příkaz **commit=y** společně s hodnotou pro parametr **buffer.** Po daném počtu bude provedeno automatické potvrzení, jak je vidět na následujících příkladech. V prvním uvedeném příkladu **import** bude proveden příkaz **commit** po každém načtení tabulky. Ve druhém příkladu uvedeném v příkladě je příkaz **commit** proveden po každém vložení 64 000 bajtů.

Imp system/heslo file=expdat.dmp

Imp system/heslo file=expdat.dmp buffer=64000 commit=Y

Hodnota buffer by měla být dostatečně velká, aby bylo možné zpracovat i ten nejdelší importovaný řádek, pokud nevíme, jakou hodnotu má nejdelší importovaný řádek, je lépe začít s nějakou pravděpodobnou hodnotou (např. 50 000) a spustíme import. Pokud bude vrácena chyba **IMP-00020**, není hodnota parametrů buffer dostatečně velká. Zvýšíme hodnotu a opakujeme import.

Při použití parametru commit=y je třeba pamatovat na to, že příkaz commit se provádí pro jednotlivá pole buffer. To znamená, že pokud import tabulky selže, je možné, že budou některé řádky tabulky již importovány a potvrzeny (commit). Následně je před dalším importem možné použít částečné načtení nebo příkaz delete.

V tabulkách se sloupci s datovými typy BFILE, LONG, LOB, REF, ROWID a UROWID se řádky vkládají individuálně. Hodnota buffer nemusí vyhovovat v řádku sloupcům LONG a LOB. Program pro import se pokusí alokovat další oblast vyrovnávací paměti, která pro něj bude potřeba [14].

5.5.1.4 Import odlišných účtů

Chceme-li přenést objekty mezi uživateli prostřednictvím exportu a importu, provedeme uživatelský export vlastníka objektů. Během importu zadáme vlastníka do klauzule fromuser a účet, který bude objekty vlastnit do klauzule touser.

Pokud tady například kopírujeme objekty uživatele THUMPER do účtu FLOWER, můžeme provést následující příkazy. První příkaz exportuje vlastníka THUMPER a druhý příkaz importuje objekty THUMPER do účtu FLOWER.

Exp system/heslo file=thumper.dat owner=thumper grants=N indexes=Y compress=Y rows=Y

Imp system/heslo file=thumper.dat fromuser=thumper touser=flower rows=Y indexes=Y

5.5.1.5 Import selhávajících struktur

Pomocí parametru rows je možné opětovně vytvořit pouze strukturu objektů bez dat v tabulce, i v případě, že byla data exportována. Parametr rows je užitečný také při po sobě jdoucích importech pro objekty, které nebyly vytvořeny při předchozím pokusu o import. Pokud při prvním importu došlo k chybám, bude pravděpodobně třeba provést více importů. Při dalším importu do stejného schématu bude obvykle třeba zadat hodnotu ignore=y a zabránit tak tomu, aby byl import ukončen z toho důvodu, že již tabulky existují.

Takový případ je vidět v následujícím příkladu. První import se pokusí zpracovat celý výstupní soubor exportu. Pokud dojde při tomto importu k chybám, bude proveden druhý pokus, při kterém se budou importovat struktury, jejichž import při prvním pokusu selhal.

Imp system/heslo file=expdat.dmp full=y commit=y buffer=64000

Při importu dojde k neopravitelným chybám u indexů po vložení dat. Nyní spustíme import podruhé s hodnotami ignore=y a rows=n.

Imp system/heslo file=expdat.dmp ignore=y rows=n buffer=64000 commit=y

Parametr ignore=y ve druhém příkazu pro program importu znamená, že se mají ignorovat všechny objekty vytvořené při prvním pokusu o import. Budou importovány pouze ty objekty, které ještě ve schématu neexistují. Úspěšné opětovné vytvoření všech databázových objektů bude pravděpodobně vyžadovat, aby bylo spuštěno více importů s parametrem rows=n [14].

5.5.1.6 Komplexní export a import dat

Zajímavou alternativou zálohování databáze, je využití tvorby vlastních sql skriptů, které zajišťují, že následná obnova databáze bude plně funkční a to se všemi nadefinovanými rolemi a právy. Budou vytvořena synonyma, ať již lokální tak i veřejná. Pokud je vše dovedeno do důsledků, uživatelé mají obnovená i původní hesla. To vše lze provést velice efektivní a efektní cestou.

Metodikem těchto postupů je Ing. Pavel Holub[26], který je zároveň i oponentem této práce. Na základě jeho odborného vedení vznikly následující skripty resp. postupy. Vzhledem k rozsahu práce, budou uvedeny jen fragmenty skriptů, které však při dosazení reálných proměnných jsou plně funkční. Nevýhodou tohoto způsobu zálohování je fakt, že takto lze obnovit databázi, z které již skripty jsou vytvořené. Tuto metodu lze tedy primárně spíše využít více k migraci databáze na jiný server či na jinou verzi databáze, než k řešení havárií. Kdo však pravidelně dávku skriptů spouští na ostrou databázi, tak společně se souborem importu – dump soubor[27] – vytváří kvalitní a spolehlivý způsob obnovy databáze. Pokud se k daným skriptům připojí i skript pro vytvoření nové databáze a uživatelů pro import, pak se jedná o komplexní řešení obnovy.

Zde je seznam sql dotazů, které vytvoří obnovovací skripty:

1. **Seznam uživatelů v databázi.**

```
SELECT ""||name||"','from sys.user$ where type#=1 AND name NOT IN ('SYS', 'SYSTEM', 'CTXSYS', 'DBSNMP', 'OUTLN', 'WMSYS')order by ctime DESC;
```

[26] Ing. Pavel Holub pracuje ve společnosti Geovap spol. s r.o. jako programátor, analytik a správce DB systémů.

[27] Dump soubor je textový soubor, obsahující všechna data, která jsou uložena v databázi, včetně řídících dat. (Uživatelská jména, hesla, názvy tabulkových prostorů, databázových souborů, názvy synonym, pohledů a v neposlední řadě i informací o primárních a cizích klíčích).

Výsledkem tohoto příkazu je ucelený seznam uživatelů, vyjma systémových uživatelů, kteří jsou vytvořeni znovu při zakládání nové databáze.

Ukázka výstupu:

('KOSOVA','CEROVSKA','ROUSAR','LEMBERKOVA','BRYCHOVA',

'PROKUPKOVA','PRAUSOVA')

2. Vytvoření uživatelů a přidělení oprávnění k připojení do databáze.

SELECT 'CREATE USER '||a.name||' identified by heslo default tablespace '||b.name||' temporary tablespace '||c.name||' profile default;'||CHR(13)||CHR(10) ||'GRANT "CONNECT" TO '||a.name||'";' FROM sys.USER$ a, sys.ts$ b, sys.ts$ c WHERE TYPE# = 1 AND a.datats# = b.ts# AND a.tempts# = c.ts# AND a.name IN ('KOSOVA');

Výsledkem je skript, který využije předchozí výstup a vytvoří příkazy pro založení jednotlivých uživatelů databáze a nagrantuje jim právo připojení k databázi.

Ukázka výstupu:

CREATE USER KOSOVA identified by heslo default tablespace USERS temporary tablespace TEMP profile default; GRANT "CONNECT" TO "KOSOVA";

3. Založení všech nesystémových[28] rolí.

SELECT 'CREATE ROLE '||name||';' FROM SYS.USER$ WHERE name NOT IN ('Nesystémové role‘) AND TYPE#=0 ORDER BY name;

Tento skript vypíše všechny nesystémové role z databáze a vytvoří z nich sekvenci pro vytvoření.

Ukázka výstupu:

CREATE ROLE BYTY_ALL; CREATE ROLE BYTY_DLUHY; CREATE ROLE BYTY_FAKTURY; CREATE ROLE BYTY_NAJEM;

[28] Při vytváření nové databáze, jsou rovněž vytvořeny i systémové role, tudíž není nutné ve skriptu systémové role vybírat a znovu vytvářet.

4. Skript pro založení veřejných synonym[29].

```
SELECT 'CREATE PUBLIC SYNONYM '||SYNONYM_NAME||' FOR
'||TABLE_OWNER||'.'||SYNONYM_NAME||';' FROM ALL_SYNONYMS WHERE
All_synonyms.OWNER = 'PUBLIC' AND all_synonyms.TABLE_OWNER IN

('KOSOVA','CEROVSKA','ROUSAR');
```

Skript vytvoří seznam veřejných synonym pro jednotlivé objekty v databázi, bez nutnosti znalosti příslušnosti objektu ke schématu.

Ukázka výstupu:

```
CREATE PUBLIC SYNONYM C_FORMTISK FOR
POPLATKY.C_FORMTISK;

CREATE PUBLIC SYNONYM C_KALENDAR FOR
POPLATKY.C_KALENDAR;

CREATE PUBLIC SYNONYM C_KATASTR FOR BYTY.C_KATASTR;

CREATE PUBLIC SYNONYM C_KDO_SCHVALIL FOR
BYTY.C_KDO_SCHVALIL;
```

5. Skript pro založení lokálních synonym.

```
SELECT 'CREATE SYNONYM '||owner||'.'||SYNONYM_NAME||' for
'||table_owner||'.'||synonym_name||';' FROM ALL_SYNONYMS WHERE
All_synonyms.OWNER <> 'PUBLIC' AND all_synonyms.OWNER IN

('KOSOVA', CEROVSKA', 'ROUSAR');
```

Skript vytvoří seznam privátních neboli lokálních synonym[30]. Tabulka ve schématu SURAD po vytvoření synonyma je pro uživatele zpřístupněna pod jeho vlastním schématem.

Ukázka výstupu:

```
CREATE SYNONYM KOSOVA.SKPRA for SURAD.SKPRA;
```

[29] Veřejné synonymum napomáhá zjednodušeně se odkazovat na veřejné objekty v databázi.
[30] Lokální synonymum napomáhá zjednodušeně se odkazovat na privátní objekty v databázi.

CREATE SYNONYM CEROVSKA.SKPRD for SURAD.SKPRD;

CREATE SYNONYM ROUSAR.SKPREP for SURAD.SKPREP;

6. Skript pro nagrantování rolí uživatelům.

SELECT 'GRANT '||granted_role|| ' TO '||grantee||';' FROM DBA_ROLE_PRIVS;

Tento skript zaručí zpřístupnění všech rolí v databázi pro určené uživatele.

Ukázka výstupu:

GRANT UAP_ALL TO GIS;

GRANT RESOURCE TO GIS;

GRANT ROB_VIDET TO GIS;

GRANT UIR_VIDET TO GIS;

7. Granty přímo do uživatelů.

SELECT 'GRANT '||privilege|| ' on '||owner||'.'||TABLE_NAME||' to '||grantee||
DECODE(grantable, 'YES',' with grant option;',' ;') FROM DBA_TAB_PRIVS
WHERE grantee IN ('KOSOVA', 'CEROVSKA', 'ROUSAR");

Tento skript vytváří oprávnění pro každého uživatele, co může s danou tabulkou dělat. Výchozí možnosti jsou select, update, insert, delete.

Ukázka skriptu:

GRANT INSERT on SURAD.SXKN_TYPRAV to KOSOVA ;

GRANT SELECT on SURAD.SXKN_TYPRAV to CEROVSKA ;

GRANT UPDATE on SURAD.SXKN_TYPRAV to ROUSAR ;

8. **Skript pro obnovu hesel uživatelů.**

SELECT 'alter user '||name||' identified by values '"'||password||'";' FROM SYS.USER$ WHERE name IN ('KOSOVA', 'CEROVSKA', 'ROUSAR', 'LEMBERKOVA');

Tento skript vrátí všem uživatelů původní hesla.

Ukázka výstupu:

alter user KOSOVA identified by values '4E507244D956C68A';

alter user CEROVSKA identified by values '4648E8B42C1C80AE';

alter user ROUSAR identified by values 'C3050HC15B2A4A71';

alter user LEMBERKOVA identified by values '2094513D3BAA846F';

Takto vytvořené skripty pak lze poměrně snadno využít k vytvoření identické databáze, respektive její obnovy.

Plán obnovy:

1. **Lze využít hotový skript nebo pomocí průvodce je nutné vytvořit novou databázi. Je vhodné, pokud se jedná o obnovu do provozního stavu, aby se název shodoval s původní databází.**
2. **V druhém kroku je nutné spustit skript, který založí všechny potřebné tabulkové prostory a založí uživatele databáze, kteří jsou vlastníci objektů v databázi (schémat) a přiřadí uživatelům oprávnění pro připojení a import dat.**

   Ukázka skriptu:

   Vytvoření tabulkového prostoru

   CREATE TABLESPACE TEST

   DATAFILE 'E:\ORADATA\DB1\TEST01.dbf' SIZE 105M
   AUTOEXTEND ON NEXT 10M MAXSIZE 2000M

   DEFAULT STORAGE (INITIAL 10K NEXT 50K MAXEXTENTS UNLIMITED PCTINCREASE 10);

Vytvoření uživatele

CREATE USER "TESTER" IDENTIFIED BY HESLO DEFAULT TABLESPACE TEST TEMPORARY TABLESPACE TEMP PROFILE DEFAULT;

GRANT "CONNECT" TO "TESTER";

GRANT "RESOURCE" TO "TESTER";

3. **Následuje spuštění výstupů ze skriptů.**
   a. Vytvoření uživatelů a přidělení oprávnění k připojení do databáze.
   b. Založení všech nesystémových[31] rolí.
4. **V tento okamžik je nutné spustit import databáze.**

   Imp SYSTEM/heslo@DB1 FILE=c:\TMP\DB.dmp fromuser=Test1, Test2 touser=Test1, Test2
5. **Po importu dat následuje spuštění dalších skriptů.**
   a. Skript pro založení veřejných synonym.
   b. Skript pro založení lokálních synonym.
   c. Skript pro nagrantování rolí uživatelům.
   d. Granty přímo do uživatelů.
   e. Skript pro obnovu hesel uživatelů.
6. **Databáze je ve funkčním stavu.**

Zajímavostí tohoto postupu je, že v literatuře není příliš rozšířen a je to spíše otázka vynalézavosti. Postup je to skutečně funkční, což už bylo v rámci instituce Magistrátu města Pardubic několikrát vyzkoušeno. Primárně je však tento postup určen k migraci databáze, jak již bylo zdůrazněno na začátku kapitoly.

[31] Při vytváření nové databáze, jsou rovněž vytvořeny systémové role, tudíž není nutné ve skriptu systémové role vybírat a znovu vytvářet.

5.5.1.7 Nasazení záloh online

Konzistentní zálohy offline je možné provádět v době, kdy je databáze vypnutá. Je však také možné provádět zálohy fyzických souborů databáze i v době, kdy je databáze otevřena a v provozu, za předpokladu, že je databáze v režimu ARCHIVELOG a záloha je provedena správně. Tyto zálohy se nazývají zálohy online.

V případě provozu databáze Oracle v režimu archivelog vytvoří proces na pozadí ARCH kopie jednotlivých souborů transakčního protokolu poté, co do nich proces LGWR přestane zapisovat. Tyto archivní soubory transakčního protokolu se obvykle zapisují na diskovou jednotku, jedná se však o operace velmi náročné na obsluhu operátora.

Chceme-li využívat možnosti režimu ARCHIVELOG, musí být nejdříve databáze nastavena do tohoto režimu. V následujícím výpisu jsou uvedeny kroky potřebné pro nastavení databáze do režimu ARCHIVELOG.

SQL> connect sys/heslo as sysdba (Přihlášení k databázi uživatelem SYS.)
SQL> startup mount db1; (Start databáze db1 do režimu mount.)
SQL> alter database archivelog; (Přepnutí databáze do stavu "Archivelog".)
SQL> alter database open; (Otevření databáze.)
Pomocí následujícího příkazu zjistíme aktuální stav ARCHIVELOG databáze.

SQL> archive log list

Databáze nastavená do režimu ARCHIVELOG zůstane v tomto režimu až do té doby, než bude nastavena do režimu NOARCHIVELOG. Místo, kde jsou uloženy archivní soubory transakčního protokolu, je určeno nastavením v souboru s databázovými parametry (init.ora). Důležité jsou zejména tyto parametry [14]:

Log_archive_dest_1 = /u1/Oracle/arch/db1/arch
Log_archive_dest_state_1 = ENABLE
Log_archive_start = TRUE

5.5.1.8 Zálohování při maximální dostupnosti 24 x 7

Jakmile je databáze spuštěna v režimu ARCHIVELOG, je možné ji zálohovat, i když je otevřená a dostupná všem uživatelům. To umožňuje udržovat databázi

k dispozici 24 hodin denně, přičemž ji však neustále můžeme zálohovat a obnovovat ze zálohy.

Přestože je možné provádět zálohy online během normální pracovní doby, je vhodnější je spíše plánovat na dobu, s co nejmenším zatížením databáze ze strany uživatelů a to z několika důvodů.

Za prvé platí, že zálohy online budou pro zálohování fyzických souborů používat příkazy operačního systému, přičemž tyto příkazy budou využívat dostupné vstupně – výstupní prostředky v systému. Za druhé platí, že při zálohování tabulkových prostorů se mění způsob, jakým jsou transakce zapisovány do archivních souborů transakčního protokolu. Po nastavení tabulkového prostoru do režimu zálohování zapisuje proces DBWR všechny bloky ve vyrovnávací paměti náležící k libovolnému souboru, který je součástí tohoto tabulkového prostoru, zpět na disk. Jakmile jsou bloky načteny zpět do paměti a následně změněny, budou zkopírovány do vyrovnávací paměti protokolů při jejich první změně.

Pokud bloky zůstanou ve vyrovnávací paměti, nebudou znovu zkopírovány do transakčního protokolu. Díky tomu bude využito značné místo v cílovém adresáři s archivními soubory transakčního protokolu.

Soubor příkazů pro zálohy online má tři části:

1. Zálohování datových souborů na základě jednotlivých tabulkových prostorů, které se skládá z těchto kroků:
    a) nastavení tabulkového prostoru do stavu zálohování,
    b) zálohování datových souborů tabulkového prostoru,
    c) uvedení tabulkového prostoru ze zálohy do normálního stavu.
2. Archivace archivních souborů transakčního protokolu, se skládá z těchto kroků:
    a) Zaznamenání informací o tom, které soubory jsou v cílovém adresáři s archivními soubory transakčního protokolu.
    b) Archivace archivních souborů a následně (volitelně) jejich odstranění nebo komprese.

3. Zálohování řídících souborů prostřednictvím příkazu **alter database backup controlfile**.

Proces zálohování za provozu je automatizován prostřednictvím nástroje **RMAN**, kterému se věnuje následující kapitola[14].

5.5.2 RMAN – Recovery manager

Od verze Oracle 8.0 je k dispozici sada nástrojů s názvem RMAN (Recovery Manager) umožňující zálohovat a obnovovat databáze ze zálohy automaticky v režimu příkazového řádku. Přestože RMAN představuje vynikající nástroj pro zabezpečení databází a zajištění úspěšného obnovení, nelze se používáním tohoto nástroje vyhnout otázkám plánování záloh [14].

Proč používat místo metod uživatelské správy právě nástroj RMAN? Nástroj RMAN poskytuje rozhraní API[32] pro správu médií a je tedy možné snadno integrovat software pro správu médií třetích stran. Při provádění záloh online nástrojem RMAN jsou za účelem dosažení konzistence dat opětovně čteny všechny fragmentované datové bloky [14].

5.5.2.1 Architektura RMAN

Nástroj Recovery Manager obsahuje čtyři součásti: spustitelné soubory RMAN, cílovou databázi, databázi obnovovacího katalogu a software pro správu medií. Z toho jsou nutné pouze spustitelné soubory a cílová databáze. Vzhledem k tomu, že nástroj RMAN ukládá automaticky svá metadata[33] v řídícím souboru cílové databáze, není obnovovací katalog třeba. Protože však používání obnovovacího katalogu skýtá různé další výhody, je doporučeno ho využívat [14].

5.5.2.2 Použití RMAN

Spustitelný soubor RMAN je obecně uložen v adresáři $ORACLE_HOME/bin v systému unix a v adresáři ORACLE_HOME/bin v prostředí Windows, avšak umístění se může měnit v závislosti na použitém operačním systému.

[32] Zkratka slov Application Programming Interface. Obvykle označuje sadu předpřipravených funkcí či objektů, které usnadňují programátorům programování.
[33] Zjednodušeně řečeno, jsou to data o datech (Data, která popisují data.).

Z příkazové řádky se k cílové databázi přihlásíme například následujícím příkazem:

rman target HR

Kde *HR* je název cílové databáze a *target* znamená řetězec připojení pro cílovou databázi.

Nástroj RMAN poskytuje ještě jednu výhodu - sekvence příkazů je možné shromáždit do jediného skriptu, který pak může naráz vykonat více příkazů najednou.

Zde je příklad zálohovacího skriptu:

rman {	
allocate channel R1 type disk format	(Alokuje kanál úložiště nazvaný R1.)
'c:\zaloha\bck_full'	(Fyzické umístění cesty kanálu R1.)
backup database;	(Příkaz pro zálohu databáze.)
release channel R1;	(Uvolnění kanálu R1.)
}	(Konec vykonávání skriptu.)

5.5.3 Offline záloha

Souborová záloha offline je fyzická záloha databázových souborů provedená po zastavení databáze prostřednictvím příkazů **shutdown normal, shutdown immediate nebo shutdown transactional.** V době, kdy je databáze vypnuta, jsou zálohovány jednotlivé soubory, jinak aktivně využívané databází. Tyto soubory představují ucelený snímek databáze tak, jak existovala v okamžiku, kdy byla vypnuta.[34]

Při zálohování offline je třeba zálohovat následující soubory:

- datové soubory,
- řídící soubory,
- aktivní soubory transakčního žurnálu.

[34] Nelze příliš spoléhat na zálohu provedenou po zadání příkazu **shutdown abort**, jelikož záloha by mohla být nekonzistentní.

Volitelně je možné zvolit zálohování souboru s inicializačními parametry databáze, a to zejména v případě, že bude taková záloha sloužit jako základ pro obnovení databáze při selhání [14].

Vzhledem k tomu, že během záloh offline dochází ke změně dostupnosti databáze, jsou obvykle zálohy plánovány na noc. Soubor s příkazy pro automatické provádění záloh by vypadal následovně [14]:

sqlplus /nolog
(S*puštění prostředí SQLPLUS.)*

conn sys/heslo as sysdba
(*Připojení k databázi jako sys dba uživatel.)*

shutdown transactional;
(*Zastavení databáze, které vyčká na dokončení všech transakcí.)*

exit;

xcopy /e $Oracle_base\Oracle*.* c:\Oratmp\e\Oracle*.*
(*Zkopírování databázových datových souborů.)*

move $Oracle_base \archiv_db_name c:\Oratmp\e\
(*Přesunutí adresáře s archivními logy do zálohovaného adresáře.)*

md $Oracle_base \archiv_ db_name
(Vytvoření nového adresáře pro archivní redology.)

xcopy /e c:\Oracle\admin\db_name*.* c:\Oratmp\c\Oracle\ db_name *.*
(*Zkopírování inicializačního souboru databáze a logů.)*

xcopy c:\Oracle\oradata\ db_name *.* c:\Oratmp\c\Oracle\oradata*.*
(*Zkopírování záložních řídících souborů a redologů.)*

xcopy c:\Oracle\ora92\database*.* c:\Oratmp\c\Oracle\ora92*.*
(*Zkopírování spfile a souboru s heslem uživatele sys.)*

sqlplus /nolog
(*Spuštění prostředí SQLPLUS.)*

conn sys/heslo as sysdba
(*Připojení k databázi jako sys dba uživatel.)*

startup;
(*Nastartování databáze kvůli okamžité dostupnosti.)*

exit;

rar.exe a -hpheslo -agDD_MM_YY -df -m0 -r -t c:\offlin~1\DB_name e:\Oratmp\
(Spuštění komprimačního programu rar, který zaarchivuje všechny vykopírované soubory databáze, vytvoří soubor, jehož jméno bude složeno z data, roku a nastaví na archiv heslo „heslo“.)

5.5.3.1 Příklad zálohy a obnovy dbs

Provozovaná aplikace	Ginis[35]
Databázová platforma	Oracle 9.2.0.8
Operační systém	Windows Server 2003 Standard

Na tomto systému bude představena zálohovací strategie, realizována pomocí exportů dat z databáze a následného obnovení dat (importu) do plné funkčnosti na nový databázový server s operačním systémem Windows 2003.

Před praktickou realizací skriptu, který by uměl automaticky provést zálohu databáze pomocí exportovací utility *exp,* bylo zapotřebí posoudit, zda plánovaný systém zálohy a obnovy postačuje. Dále bylo nutné zjistit, zda riziko ztráty dat jednoho dne není příliš vysoké a jaké ztráty by mohly nastat.

a) Pomocí utility *exp*, lze velmi pohodlně vyexportovat data ze všech tabulek z databáze. Tato metoda, jak již bylo řečeno v předchozích kapitolách, je velmi rychlá, bohužel problém může nastat při následném importu dat, s tzv. referenční integritou[36]. Problém by mohl nastat, pokud by se v databázi vyskytovalo více vlastníků jednotlivých dat, resp. tabulkových prostorů, s návazností jeden na druhého. Druhý problém spočívá v tom, že metoda uživatelského exportu vyžaduje vyrobit databázovou strukturu (fyzickou strukturu), která je zapotřebí pro import dat.

b) Export dat pomocí utility *exp* umožňuje provádět zálohování bez nutnosti zastavení databáze, čili umožňuje zálohování online. Je zde jisté riziko, že pokud uživatel provede změnu v databázi po vyexportování dat z tabulky, změny nebudou v záloze obsaženy, proto je vhodné provádět zálohování pomocí exportu jednou denně (aby nedocházelo k nadměrnému zatěžování

[35] Ginis – Software společnosti Gordic s.r.o. (Ekonomické moduly, modul spisové služby).
[36] Referenční integrita – definuje se jako cizí klíč, který musí ukazovat na nějaký existující klíč.

systému) nejlépe v nočních hodinách, kdy už není předpoklad, že některý z uživatelů bude modifikovat data v databázi nebo použít přepínače **consistence=y**. V obou případech je však výhodnější provádět zálohy v době, kdy k databázi v ideálním případě není připojen žádný uživatel, který by mohl měnit data, aby export byl plně konzistentní.

c) Z výše uvedeného vyplývá, že zálohování dat pomocí exportovací utility *exp* s sebou přináší určitá rizika spojená především s jednodenní ztrátou dat, pokud by pád databáze nastal ve večerních hodinách před vykonáním zálohy. Poslední funkční záloha by byla z předešlého dne z večera. Jelikož se však nejedná o systém, který vyžaduje nepřetržitou funkčnost a ztráta dat za jeden den je označena jako únosné riziko, je tento model zálohování pro tento aplikační software vyhovující. Je dobré alespoň databázi uvést do tzv. archivního režimu, který napomáhá udržet konzistenci databáze při pádech a je součástí vnitřních automatických obnovovacích mechanismů, kterými Oracle disponuje. Zálohování metodou EXP je rovněž využito i při výkonnostních testech.

5.5.4 Zálohovací skript

5.5.4.1 Nastavení NLS prostředí

Po zajištění všech předchozích podmínek pro export nastavíme prostředí pro export a spustíme exportovací nástroj s parametry:

Nastavení NLS prostředí databáze (*Jedná se o nastavení národního prostředí.).*
MODE CON CODEPAGE SELECT=1250 > NUL
SET NLS_DATE_FORMAT=YYYY-MM-DD HH24:MI:SS
SET NLS_DATE_LANGUAGE=CZECH
SET NLS_ISO_CURRENCY=CZECH REPUBLIC
SET NLS_NUMERIC_CHARACTERS=,.
SET NLS_SORT=BINARY
SET NLS_LANG=CZECH_CZECH REPUBLIC.EE8MSWIN1250
(*V systému unix je třeba po nastavení parametrů tyto parametry vyexportovat do jednotlivých relací.)*
Nastavení cest Oracle.
(*Je nutné nastavit cesty k jednotlivým adresářům Oracle, které jsou nezbytné při spouštění skriptů.)*
@SET ORAORAPATH=C:\Oracle\Ora90
@SET ORASCRIPTSPATH=C:\Oracle\scripts

@SET ORAFULLEXPORTPATH=C:\Oracle\FullExport
@SET ORAUTILUSENET=yes
@SET ORAUTILEXP=exp
@SET ORAUTILIMP=imp
@SET ORAUTILSQLPLUS=sqlplus
@rem SET ORAUIRADR=no
@rem SET ORAMSM=no

5.5.4.2 Příkaz exportu

exp system/heslo file=c:\zaloha\db1.dmp owner=dbowner log= c:\zaloha\DB1\db1.log
(*Tento příkaz provádí logickou zálohu ve formě exportu do souboru C:\Zaloha\DB1\db1.dmp uživatele dbowner a zapisuje o celém exportu log soubor C:\Zaloha\DB1\db1.log.)*

Je vytvořen datový soubor c:\zaloha\db1.dmp a soubor logu c:\zaloha\DB1\db1.log, z kterého je možné vyčíst, zda export proběhl úspěšně.

5.5.4.3 Přenos na úrovni file systému (FS)

V dalším kroku je zapotřebí soubor db1.dmp z Windows serveru 2003 A, přenést na server Windows 2003 B, což lze zrealizovat pomocí FS (file systémového) přenosu, resp. skriptu:

Obsah skriptu down.bat:
ren \\BackupPC\Zaloha_db$[37]*.dmp *.del >c:\zaloha\copy.log[38]
ren \\ BackupPC \Zaloha_db$*.log *.de >>c:\zaloha\copy.log
(Pomocí příkazů ren se přejmenují soubory se zálohou a logem z předchozího dne.)

copy c:\zaloha*.* \\BackupPC\Zaloha_db$*.* >>c:\zaloha\copy.log
(Pomocí příkazu Copy jsou zkopírovány nové sobory zálohy a logu do cílového umístění.)

del \\ BackupPC \Zaloha_db$*.del >>c:\zaloha\copy.log
del \\ BackupPC \Zaloha_db$*.de >>c:\zaloha\copy.log
(Následuje odstranění starých souborů zálohy a logu.)

5.5.4.4 Obnova databáze na serveru Windows 2003 Std.

1. Nástrojem **oradim** je vytvořena nová **instance** databáze DB1, která převezme parametry z **inicializačního souboru** initDB1.ora[39]

oradim –new –sid **DB1** pfile=c:\Oracle\admin\DB1\pfile\initDB1.ora

[37] Konvence $ je v tomto případě využita z důvodu skrytého sdílení adresáře, které pro běžné uživatele není viditelné.
[38] Vše, co se během přenosu stane, se loguje vždy do textového souboru copy.log.
[39] initDB1.ORA je iniciační soubor databáze.

2. V prostředí SQLplus se nastaví parametry NLS prostředí pro databázi.
(*Jedná se o nastavení národního prostředí.)*
SET NLS_DATE_FORMAT=YYYY-MM-DD HH24:MI:SS
SET NLS_DATE_LANGUAGE=CZECH
SET NLS_ISO_CURRENCY=CZECH REPUBLIC
SET NLS_NUMERIC_CHARACTERS=,.
SET NLS_SORT=BINARY
SET NLS_LANG=CZECH_CZECH REPUBLIC.EE8MSWIN1250

3. Další krok je spuštění skriptu na vytvoření databáze
CREATE DATABASE DB1
(Vytvoří databázi s názvem DB1.)

USER SYS IDENTIFIED BY heslo
(*Nastaví heslo uživatele SYS na heslo.)*

USER SYSTEM IDENTIFIED BY heslo
(*Nastaví heslo uživatele SYSTEM na heslo.)*

LOGFILE
(Založí 4 redo logy o velikosti 8MB.)
'db1./redo01.log' SIZE 8M,
'db1./redo02.log' SIZE 8M,
'db1./redo03.log' SIZE 8M,
'db1./redo04.log' SIZE 8M

MAXLOGFILES 16
(Nastaví maximální počet redologů na 16.)

MAXLOGMEMBERS 4
(Nastaví maximální počet skupin redologů na 4.)

MAXLOGHISTORY 1
(Specifikuje maximální počet archivních redologů pro obnovu RAC.)

MAXDATAFILES 60
(Specifikuje maximální počet datových souborů.)

MAXINSTANCES 1
(Specifikuje maximální počet instancí jedné databáze.)

NOARCHIVELOG
(Specifikuje, že nebude použit archivní režim[40].)

CHARACTER SET EE8MSWIN1250

[40] Noarchivelog – je to výhodnější databázi uvést do archivního režimu až po importu databáze z důvodů zbytečného archivování redologů během importu).

(Specifikuje kódování ukládaných dat do databáze.)
NATIONAL CHARACTER SET UTF8
(Specifikuje kódování národního prostředí.)

DATAFILE
(Vytváří základní datový soubor System.)
'db1./sys_dct1.dbf' SIZE 200M AUTOEXTEND ON NEXT 100M MAXSIZE 2000M

DEFAULT TEMPORARY TABLESPACE TEMP TEMPFILE
(Vytváří základní datový soubor Temp.)
'db1./sys_tmp1.tmp' SIZE 25M AUTOEXTEND ON NEXT 25M MAXSIZE 2000M EXTENT MANAGEMENT LOCAL

UNDO TABLESPACE RBS DATAFILE
(Vytváří základní datový soubor Undo tablespace.)
'db1./sys_rbs1.dbf' SIZE 200M AUTOEXTEND ON NEXT 100M MAXSIZE 2000M EXTENT MANAGEMENT LOCAL;

4. Dále se vytvoří tabulkový prostor, pro import datových struktur uživatele dbowner.

CREATE TABLESPACE DB_DATA
(Vytvoří datový soubor o velikosti 500 MB, který bude rozšiřitelný na 2 GB a dokáže sám alokovat potřebné místo.)
DATAFILE 'DB1./db_dat1.dbf' SIZE 500M AUTOEXTEND ON NEXT 250M MAXSIZE 2000M
EXTENT MANAGEMENT LOCAL AUTOALLOCATE
SEGMENT SPACE MANAGEMENT AUTO;

5. Po vytvoření databáze je nutno spustit z prostředí sqlplus následující systémové skripty, které vytvoří systémové pohledy a synonyma.

catalog.sql
catproc.sql
(*Oba skripty se nacházejí v adresáři Oracle_Home\rdbms\.)*

6. Vytvoří se uživatel, který bude vlastníkem objektů, a budou mu nastavena patřičná oprávnění pro import databáze.

CREATE USER "DBOWNER" IDENTIFIED BY "heslo"
(*Vytvoř uživatele Dbowner a nastav heslo na“ heslo“.)*
DEFAULT TABLESPACE DB_DATA
(*Jeho domovský tabulkový prostor bude DB_DATA.)*

TEMPORARY TABLESPACE TEMP;
(*Dočasný tabulkový prostor bude TEMP.)*

Veškerá nastavování práv se budou týkat uživatele DBOWNER!

GRANT DBA TO "DBOWNER";
(*Nastaví roli DBA, která umožňuje nejvyšší oprávnění v databázi.)*

GRANT UNLIMITED TABLESPACE TO "DBOWNER";
(Nastaví uživateli bezlimitní tabulkové prostory.)

GRANT SELECT ANY TABLE TO "DBOWNER";
(Nastaví oprávnění, provádět příkaz Select na kterékoli tabulce.)

GRANT INSERT ANY TABLE TO "DBOWNER";
(Nastaví oprávnění vkládat data do jakékoli tabulky.)

GRANT UPDATE ANY TABLE TO "DBOWNER"
(Nastaví oprávnění aktualizovat data v jakékoli tabulce.)

GRANT DELETE ANY TABLE TO "DBOWNER";
(Nastaví oprávnění mazat data v jakékoli tabulce.)

GRANT EXECUTE ON SYS.DBMS_PIPE TO "DBOWNER";
(Nastaví oprávnění spustit systémovou proceduru k vytvoření rour.)

GRANT EXECUTE ON SYS.DBMS_SQL TO "DBOWNER";
(Nastaví oprávnění spustit systémovou proceduru pro správu databáze.)

GRANT EXECUTE ON SYS.DBMS_SYS_SQL TO "DBOWNER";
(Nastaví oprávnění spustit systémovou proceduru pro správu databáze.)

GRANT EXECUTE ON SYS.DBMS_APPLICATION_INFO TO "DBOWNER";
(Nastaví oprávnění spustit systémovou proceduru pro správu databáze.)

GRANT EXECUTE ON SYS.DBMS_UTILITY TO "DBOWNER";
(Nastaví oprávnění spustit systémovou proceduru pro správu databáze.)

GRANT EXECUTE ON SYS.DBMS_OUTPUT TO "DBOWNER";
(Nastaví oprávnění spustit systémovou proceduru pro správu databáze.)

GRANT EXECUTE ON SYS.DBMS_STATS TO "DBOWNER";
(Nastaví oprávnění spustit systémovou proceduru pro provádění statistik.)

GRANT CREATE SEQUENCE TO "DBOWNER";
(Nastaví oprávnění vytvořit sekvenci do databáze.)

GRANT CREATE USER TO "DBOWNER";
(Nastaví oprávnění vytvořit nového uživatele.)

GRANT ALTER USER TO "DBOWNER";
(Nastaví oprávnění změnit nastavení uživatele.)

GRANT CONNECT TO "DBOWNER" WITH ADMIN OPTION;
(Nastaví oprávnění připojit se uživatelem DBOWNER jako ADMIN databáze.)

GRANT ANALYZE ANY TO "DBOWNER";
(Nastaví oprávnění analyzovat data.)

GRANT SELECT ANY DICTIONARY TO "DBOWNER";
(Nastaví oprávnění provést Select nad systémovými pohledy.)

GRANT CREATE PUBLIC SYNONYM TO "DBOWNER";
(Nastaví oprávnění vytvářet veřejná synonyma.)

GRANT DROP PUBLIC SYNONYM TO "DBOWNER";
(Nastaví oprávnění mazat veřejná synonyma.)

GRANT ALTER ANY PROCEDURE TO "DBOWNER";
(Nastaví oprávnění systémově měnit jakékoli procedury.)

GRANT ALTER ANY TRIGGER TO "DBOWNER";
(Nastaví oprávnění systémově měnit jakékoli Triggery.)

GRANT ALTER ANY TABLE TO "DBOWNER";
(Nastaví oprávnění systémově měnit jakékoli tabulky.)

GRANT ALTER ANY SEQUENCE TO "DBOWNER";
(Nastaví oprávnění systémově měnit jakékoli sekvence.)

7. Je spuštěn import[41] databáze.

IMP SYSTEM/heslo file=db1.dmp log=db1import.log fromuser=DBOWNER touser=DBOWNER analyze=no

8. Po importu je databáze připravena k použití.

5.5.5 Vliv hardwarových prostředků na export a import databáze

Velmi diskutovanou záležitostí, jež se týká databázových systémů, je vliv hardwarových prostředků na rychlost celé databáze. Při běžném provozu není databázový systém zatěžován natolik, aby rozdíly v hardwaru byly natolik zřejmé. Databázový server však musí splňovat určitá kritéria, související s rychlostí CPU,

[41] Utilita imp čte jednotlivé řádky ze souboru db1.dmp, jejichž vlastník je DBOWNER a ukládá je do příslušných řádků tabulek, log ukládá do souboru db1.log, před jednotlivým importem neanalyzuje tabulky.

s velikostí a rychlostí paměťových modulů RAM, s velikostí a rychlostí pevného disku a dalších, avšak již ne tolik důležitých, kritérií.

Aby bylo možné posoudit, které prostředky hardwaru jsou důležité pro provoz databázového systému, bylo by nutné provést velice mnoho benchmarkových testů, které by byly úzce zaměřeny na jednotlivé oblasti zkoumání vlivů. To však není záměrem této práce. Cílem této práce je shrnutí zkušeností pro informatika, který bude databázi administrovat a spravovat. Pro tyto účely se jeví jako nejvýznamnější příklad, výkon databáze při zálohování a obnově databáze.

Výchozí podmínky testu:

1) Software – Databáze Oracle 9.2.0.8, velikost obsazených tabulkových prostorů 20 GB, vlastníkem databázových prostorů je pouze jeden uživatel, parametry v inicializačním souboru Init.ora jsou po celou dobu neměnné. Operační systém Windows Server 2003 Standard SP2.
2) Hardware – PC HP 6200 – Intel Pentium 2,4 GHz core duo 2, 2 GB RAM DDR2 800 MHz, HDD RAID 1[42] 120 GB WD 7200 ot./min
3) Metoda exportu dat - EXP[43], metoda následného importu dat - IMP[44]

Výsledkem testů je čas, měřený v minutách, potřebný k vykonání exportu dat a čas potřebný k vykonání importu dat za různých hardwarových konfigurací.

[42] Hardwarové zrcadlení disků. Obsah se současně zaznamenává na dva disky. V případě výpadku jednoho disku se pracuje s kopií, která je ihned k dispozici.
[43] Více v kapitole 5.5.1.1.
[44] Více v kapitole 5.5.1.2.

1. První test.

Tabulka 4 - Podmínky testu č.1, zdroj: [vlastní].

CPU	2,4 GHz, Intel pentium Core duo 2
Paměť RAM	2 GB, DDR2 800 Mhz
HDD	120 GB RAID 1, 7200 ot./min
Operační systém	Windows Server 2003 Eng, STD, SP2
Verze databáze	Oracle 9.2.0.8

Tabulka 5 - Výsledky výchozích podmínek testu č.1, zdroj: [vlastní].

Pořadí měření	1	2	3	4	5	6	7	8	9	10
Záloha/min	22,4	22	22,1	21,5	21,5	21,4	21,5	21,5	21,1	21,1
Obnova[45]/min	67,2	68	67,1	67,4	67	68,4	69,1	68,5	68,5	68,5

2. Druhý test. Při tomto testu je 120 GB RAID 1 7200 ot./min nahrazen pevným diskem WD Raptor 72 GB, 10 000 ot./min. Ostatní konfigurace je shodná s konfigurací z testu číslo 1.

Tabulka 6 - Podmínky testu č.2, zdroj: [vlastní].

Předpoklad:	Závislost zálohy a obnovy databáze na rychlosti HDD.
CPU	2,4 GHz, Intel pentium Core duo 2
Paměť RAM	2 GB, DDR2 800 Mhz
HDD	**72 GB, 10 000 ot/min, WD**
Operační systém	Windows Server 2003 Eng, STD, SP2
Verze databáze	Oracle 9.2.0.8

Tabulka 7 - Výsledky testu č.2, zdroj: [vlastní].

Pořadí měření	1	2	3	4	5	6	7	8	9	10
Záloha/min	16,3	16,2	16,2	16,1	15,9	15,9	15,8	15,9	15,7	15,8
Obnova/min	50,3	51	50,8	51,1	49,8	48	50	51,4	51,5	51,1

[45] Příkaz je rozepsán v kapitole 5.5.4.4, bod 7.

3. Třetí test. Při tomto testu je 2 GB modul RAM doplněn ještě jedním 2 GB modulem stejných parametrů. Ostatní konfigurace je shodná s konfigurací z testu číslo 1.

Tabulka 8 - Podmínky testu č.3, zdroj: [vlastní].

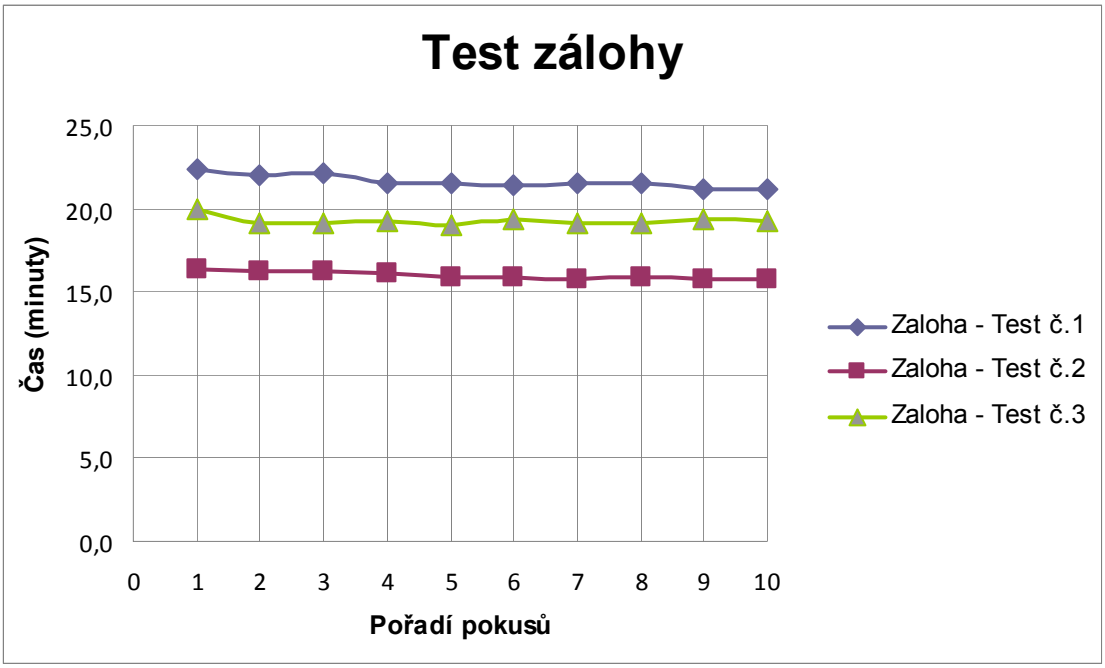
Předpoklad:	Závislost zálohy a obnovy databáze na velikosti RAM.
CPU	2,4 GHz, Intel pentium Core duo 2
Paměť RAM	**4 GB, DDR2 800 Mhz**
HDD	120 GB RAID 1, 7200 ot./min
Operační systém	Windows Server 2003 Eng, STD, SP2
Verze databáze	Oracle 9.2.0.8

Tabulka 9 - Výsledky testu č.3, zdroj: [vlastní].

Pořadí měření	1	2	3	4	5	6	7	8	9	10
Záloha/min	19,9	19,1	19,1	19,2	19	19,3	19,1	19,1	19,3	19,2
Obnova/min	64,4	64,5	64,5	65,1	64,9	64,8	64,8	65,1	65	64,9

5.5.5.1 Vyhodnocení testů

Z výše uvedených výsledků je patrno několik zjištění. Aniž by bylo nutné využít metod statistické analýzy, ze srovnání testu číslo 1 a testu číslo 2 jasně vyplývá, že rychlost disku má významný vliv na dobu jak exportu dat, tak na dobu importu dat. Pro lepší přehlednost poslouží následující graf.

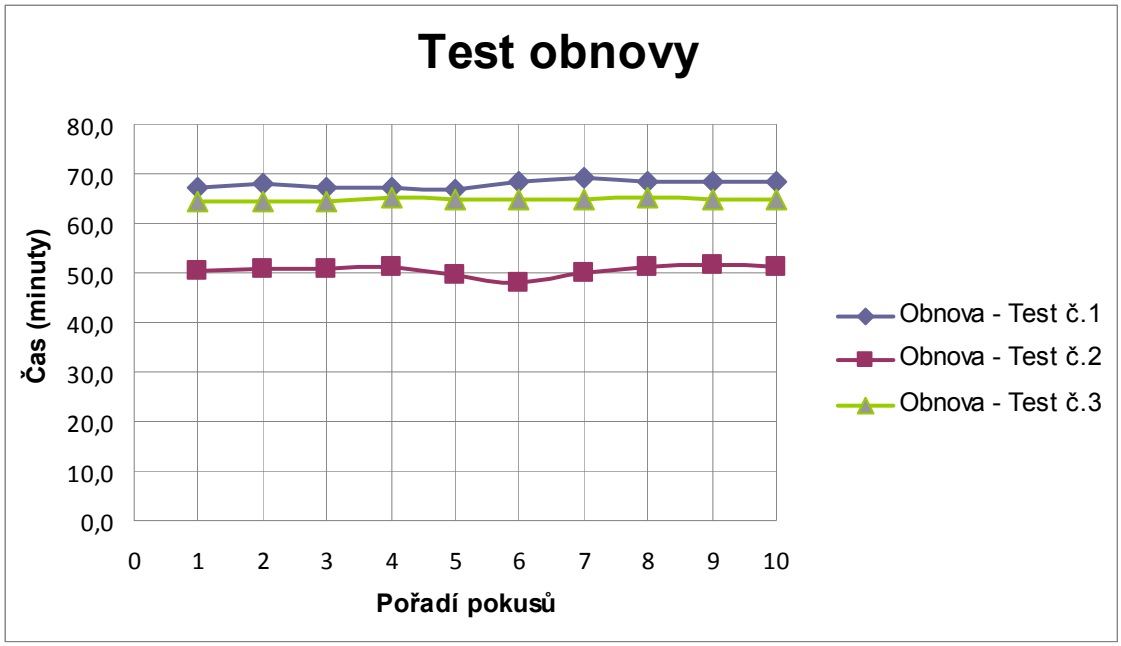


Graf 1 - Srovnání zálohy, zdroj: [vlastní].

Z grafu vyplývají tři skutečnosti. První skutečnost plně ukazuje na závislost doby exportu na rychlosti pevného disku, kdy rozdíl v časech se pohybuje řádově v několika minutách. Z grafu je rovněž patrné i zrychlení exportu po přidání dalšího paměťového modulu, zrychlení exportu však není tak výrazné, jako v případě výměny pevného disku.

Druhá skutečnost, která vyplývá z grafu je více připisována vlastnostem databázového systému Oracle. Je zřejmé, že čas, který byl u všech testů zapotřebí k vykonání exportu, byl u prvních pokusů nejvyšší a směrem k dalším pokusům se zmenšoval. Je to zapříčiněno vlastností systému Oracle, který příkazy a jejich výsledky ukládá do své alokované paměti a při opětovném zavolání příkazu je jeho provedení rychlejší. Jelikož objem alokované paměti Oraclu není neomezený, časový rozdíl mezi prvním a posledním pokusem činí řádově minutu.

Při porovnávání obnovy dat jsou rozdíly v použitém hardwaru ještě výraznější.



Graf 2 - Srovnání obnovy databáze, zdroj: [vlastní].

V tomto případě lze na vyjádření grafu jasně vidět, jak při použití rychlejšího pevného disku, rychlost celého importu roste. Z testů lze vyvodit, že nejlepší investice do hardwaru lze udělat zakoupením výkonných a rychlých pevných disků. Při obnově dat zde není patrný „fenomén“ z exportu dat, který by zapříčiňoval klesající čas obnovy mezi pokusy. Je to zřejmé. Databáze vzniká a všechny informace jsou při každém pokusu do databáze znovu importovány. Při exportu dat se jedná stále o jednu databázi. Pokud bychom chtěli odstranit odchylku způsobenou v případě exportu dat systémovým katalogem (alokací paměti), bylo by nutné databázi vždy odstranit a znovu vytvořit. Z časového hlediska to však nebylo možné.

Všechny testy byly prováděny za předpokladu, že parametry databáze zůstanou nezměněny. Na tomto místě je však důležité poznamenat, že zejména při pokusu s přidáním paměti, pokud by se navýšila velikost paměťového oddílu v databázi Oracle nazvaná Buffer cache[46], minimálně u exportu by byl výrazný nárůst rychlosti zpracování. Nastavením paměťových oddílů v databázi Oracle se zabývá oblast ladění pro provoz databáze, to už však není cílem práce.

[46] Database buffer cache je část paměti databáze, která udržuje kopie datových bloků načtených z datových souborů. Všechny uživatelské procesy souběžně připojené k instanci db database buffercache sdílejí přístup [14].

5.6 NÁVRH ŘEŠENÍ PRO INSTITUCI VEŘEJNÉ SPRÁVY

Stejně jako pro Magistrát města Pardubic, tak i pro ostatní úřady veřejné správy je důležitá bezpečnost dat a vysoká dostupnost jejich databázového systému. Nutnost zálohování dat a jejich obnova je vedle funkčnosti databázového systému tím nejdůležitějším požadavkem. Objem dat a nároky kladené na aplikace zpracovávající data neustále rostou.

5.6.1 Výběr databázové technologie

Při rozhodování představuje důležitý informační kanál dodavatel daného produktu. Nicméně i dodavatel se snaží prodat a ne vždy jsou jim podané údajezcela pravdivé. Nemusí být sice přímo lživé, ale pravda v nich může zaznět pouze částečně. Je obecně známo, že nejen ve firmách, ale i ve státní sféře občas rozhodují o nákupech lidé, kteří nejsou z technického hlediska nákup schopni provést.

Lze konstatovat, že trh s databázovými technologiemi má v určitých oblastech podobné zvláštnosti, jako trh s jiným zbožím srovnatelné cenové hladiny. V reklamních materiálech a studiích se používají podobná přirovnání. Tudíž i v oblasti databázových technologií se vyplácí být prozíravý.

Při výběru databázového systému je zcela neocenitelná osobní zkušenost, popřípadě zkušenost zprostředkovaná třetí firmou, které důvěřujeme. Užitečné jsou rovněž i nezávislé reference, ideálně od partnerských úřadů. Odpovědného pracovníka by přitom měla zajímat také záporná kritika, i když se může zdát, že z jeho pohledu není důležitá.

Databázové technologie reprezentují rozsáhlou oblast, ve které najdou uplatnění všichni lidé, kteří potřebují nějakým způsobem chránit, ukládat a zpracovávat data. Díky sílící konkurenci mezi databázovými platformami a jejich výrobci a rovněž neustálému vývoji v databázových technologiích, lze neustále požadavky zvyšovat. Pozitivní je, že ze všech sílících tlaků vždy nejvíce získává zákazník.

5.6.2 Kritéria výběru DBS platformy

V rámci výběru vhodné databázové platformy lze jednotlivé požadavky rozdělit do řady oblastí. Možné dělení může vypadat následujícím způsobem:

- Bezpečnost a ochrana – za naprosto stěžejní patří bezpečnostní kritéria. Nejde tedy pouze o samotné zabezpečení dat proti zneužití, ale rovněž o zabezpečení zálohování a zajištění vysoké dostupnosti. Toto kritérium považuji za nejdůležitější a problematice zálohování je proto věnována celá kapitola 5.

- Hardwarové požadavky na systém – do této kategorie patří především kritéria týkající se systémových nároků, jako je velikost fyzické operační paměti, rychlost procesoru, konstrukce diskového pole, a rovněž i zálohovací mechanismus.

- Škálovatelnost – stává se důležitým mezníkem, který je stěžejní podmínkou při nákupu databázového systému. Do pojmu škálovatelnost patří také schopnost neztrácet výkon ani při stálém nárůstu objemu zpracovávaných dat.

- Lokalizace systémů – problém lokalizace je neustále aktuální, zejména s ohledem na česká specifika. Díky globalizaci je nutné, aby databázové systémy pracovaly s podporou více jazyků zároveň. To bývá někdy podceňováno, i když je známo, že řada společností operuje na větším území, než je daný stát. Jelikož se snaží systémově integrovat, databázová prostředí musí procházet přes různé lokalizace.

- Podporované platformy – tato kategorie se týká požadavků na operační a hardwarové platformy, a to od serveru přes střední vrstvy až po klienty a vývojová prostředí. Nelze se pouze zaměřit na programové vybavení, ale důležitá je také hardwarová a komunikační infrastruktura. V dnešní době jsou nejčastější řešení od společnosti HP - Proliant, jehož základem je operační systém Unix resp. Linux nebo Microsoft Windows server 2003. Druhým častým řešením jsou komplexní systémy od firmy Dell. Je to poměrně finančně náročná záležitost, vyplatí se tehdy, pokud necháme technické parametry na odbornících.

- Data – podpora dotazovacího jazyka SQL, komunikačních rozhraní, datových typů. To vše jsou jedny z nejdůležitějších kritérií, která jsou velmi závislá na zkušenostech implementátora. V dnešní době většina databázových systémů

plně vyhovuje standardům SQL a jejich otevřenost umožňuje provádět různá rozhraní mezi nimi.

- Otevřenost a standardy – dodržování standardů a otevřenost systému vůči okolí je v současné době již nutností, která nechybí u žádného většího databázového systému. Proto všechny databázové systémy zkoumané v této práci splňují požadavky na otevřenost systému, jelikož tento požadavek nabývá díky systémové integraci na významu.

- Analytické možnosti – mezi nejvýznamnější analytické možnosti lze zařadit data warehousing[47] či datamining[48]. Datové sklady umožňují provádět díky analytickým nástrojům předmětové analýzy. Tyto analýzy slouží pro systémy na podporu rozhodování.

Uvedený seznam není jistě kompletní, lze však s jistotou konstatovat, že pokud se bude databázový systém vybírat podle výše zmíněných kritérií, bude plně vyhovovat jako základ každého informačního systému.

Tabulka číslo 10, uvádí dvacet databázových platforem, které lze vzít při výběru vhodného prostředí do úvahy. Pokud však je třeba budovat robustní systém, výběr se omezuje na 5 platforem uvedených ve třetí kapitole.

[47] Datové sklady.
[48] Dolování dat.

Tabulka 10 - Vhodné databázové platformy.[49]

Název	Verze	Dodavatel	Web	APP1	APP2	APP3	APP4	Cena jednouživatelská	Cena 50 uživatelská
Adabas	7.4.3	Software AG	http://www.softwareag.com	1	4	4	3	jednotky tisíc	desítky/stovky tisíc
Advantage CA-IDMS/DB Database	2.6	Computer Associates	http://www.ca.com	1	3	5	5	jednotky tisíc	desítky/stovky tisíc
Caché	5.0	InterSystems	http://www.intersystems.cz	2	4	5	4	jednotky tisíc	desítky/stovky tisíc
DB2 Universal Database Version 8	8.2	IBM	http://www.ibm.com	2	4	5	5	zdarma	desítky/stovky tisíc
Firebird	1.5	nezávislí vývojáři	http://www.firebirdsql.org	1	3	4	3	jednotky tisíc	zdarma
Informix Dynamic Server	9.4	IBM	http://www.ibm.com	1	4	5	5	jednotky tisíc	desítky/stovky tisíc
Ingres	r3	Computer Associates	http://www.ca.com	1	4	3	3	zdarma	zdarma
InterBase	7.1	Borland	http://www.borland.com	1	3	4	3	jednotky tisíc	destíky tisíc
JDataStore	7	Borland	http://www.borland.com	1	4	3	1	jednotky tisíc	jednotky/desítky tisíc
MaxDB	7.5	MySQL AB	http://www.mysql.com	1	4	3	1	zdarma	zdarma
MS Access	2003	Microsoft	http://www.microsoft.cz	5	3	1	1	jednotky tisíc	nemá smysl
MS SQL Server	2000	Microsoft	http://www.microsoft.cz	2	3	5	4	jednotky tisíc	desítky/stovky tisíc
MySQL	4.1	MySQL AB	http://www.mysql.com	1	4	3	2	zdarma	zdarma
Oracle Database	10g	Oracle	http://www.oracle.com	1	3	5	5	jednotky tisíc	desítky/stovky tisíc
PostgreSQL	7.4.6	nezávislí vývojáři	http://www.postgresql.org	1	4	3	2	zdarma	zdarma
Progress RDBMS	10	Progress Software	http://www.progress.com	1	3	5	4	jednotky tisíc	desítky/stovky tisíc
Souborový formát DBF (s adekvátním vývojovým nástrojem)	4	--	--	4	1	1	1	zdarma	zdarma
Sybase Adaptive Server Enterprise	12.5	Sybase	http://www.sybase.com	1	2	5	5	jednotky tisíc	desítky/stovky tisíc
Sybase SQL Anywhere Studio	9	Sybase	http://www.sybase.com	4	5	3	1	jednotky tisíc	nemá větší smysl
Tamino XML Server	4.2	Software AG	http://www.softwareag.com	1	5	4	2	jednotky tisíc	desítky/stovky tisíc

APP1 – jednoduchá aplikace; APP2 – střední aplikace; APP3 – vyspělá aplikace; APP4 – nejvyspělejší aplikace (1-5, 5 nejvýhodnější)

[49] Zdroj: http://www.dbsvet.cz/storage/dbs.pdf.

Řešení pro středně velký úřad veřejné správy je založeno na mé zkušenosti jako implementátora a administrátora relačního databázového systému Oracle v instituci Magistrátu města Pardubic. Podotýkám, že to není jediné správné řešení, nýbrž doporučení, vyplývající ze závěrů, které během své praxe vyvozuji a snažím se je uplatňovat a zobecňovat.

Oracle vyhovuje všem výše popsaným kritériím. Jeho „konstrukce, povaha a škálovatelnost“ umožňuje vysokou flexibilitu celého systému a je zárukou otevřenosti pro aplikace třetích stran. Oracle je robustní a stabilní systém s velmi promyšlenými technikami zálohování a obnovy. Pro Oracle hovoří i systém podpory zvaný Metalink, který je celosvětově integrovaný a garantuje do jedné hodiny od nahlášení problému pomoc ve formě konzultace po telefonu předních světových analytických špiček Oracle nebo přímo umožňuje správu a pomoc přes speciálně vyvinutý systém vzdálené plochy.

Bezpečnost je v Oracle prvořadým úkolem. Snadná správa administrace, úměrně složitá schopnost nasazovat nové aplikace tak, aby vyhovovaly všem standardům, které Oracle respektuje a dodržuje, je samozřejmostí a komfortem, který Oracle poskytuje.

Zřejmě jedním z největších důvodů, proč zvolit Oracle je možnost zálohování a následné zpětné obnovy, kdy každá minuta je drahocenná a pokud by to mělo znamenat nejistý výsledek, nemělo by smysl o databázové platformě vůbec uvažovat. Systém zálohování skýtá mnoho variant, které lze kombinovat k naprosté dokonalosti, kdy nemusí být přerušen provoz databáze a i v případech havárie serveru je díky nové technologii grid pravděpodobnost výpadku malá.

5.6.3 Odpovídající návrh řešení

Konkrétní návrh řešení je odvozený od konfigurace databázového serveru v instituci Magistrátu města Pardubic a vychází z požadavků kladených na bezproblémové fungování databáze Oracle.

Tabulka 11 - Hardware + operační systém, zdroj: [vlastní].

Server	HP ProLiant ML 370 G5
Procesor	Intel Xeon X5450 4 jádra, 3.00 GHz / 1333 MHz
Paměť	4 GB (8 slotů, max. 2GB modul)
Interní zařízení pro ukládání	Řadič HP Smart Array P400/512 MB BBWC (pole RAID 0/1/1+0/5/6)
Síťová karta	NC373i PCI-X 1Gb
Operační systém	Microsoft Windows server 2003 Std.
Zdroj	800W, redundantní

Konfigurace diskového pole:

2 x logický disk – 1. systémový (mirror 2 x 300 GB – 15 000 ot/min)

– 2. datový 6 x 300 GB – 15 000 ot/min – Raid 5

Konfigurace databáze Oracle:

Tabulka 12 - Paměť, zdroj: [vlastní].

SGA pamět	905 MB
- Shared pool size	256 MB
- Buffer cache	504 MB
- Large Pool size	8 MB
- Java Pool size	16 MB
PGA paměť	150 MB

Paměť je konfigurována dle doporučení Oracle, velikost SGA paměti + velikost PGA paměti + paměť kterou využívá operační systém, by neměla překročit velikost fyzické paměti.

Tabulka 13 - Databázové soubory, zdroj: [vlastní].

Systémové a datové tablespace	2. datový disk – Raid5
Redologs – 2 skupiny (3 členové)	1. skupina – umístění 1. systémový mirror 2. skupina – umístění 2. datový disk Raid5
Control files (3 identické soubory) metoda zrcadlení	1. Control file umístěn na 1. systémový disk – mirror 2. Control file umístěn na 2. datovém disku – Raid5

6 ZÁVĚR

Hlavním cílem práce bylo posoudit z různých hledisek (výkon, bezpečnost, stabilita), zda nasazení DBS Oracle je pro konkrétní úřad veřejné správy vhodnou volbou. Zvláště byl cíl práce zaměřen na zálohování a obnovu databáze.

Dílčím cílem této práce bylo usnadnit specialistům IT, kteří pracují na úřadech veřejné správy, rozhodování, který databázový systém je pro jejich úřad vhodný. Orientace mezi databázovými systémy je velmi obtížná a řada pracovníků úřadů, včetně specialistů IT, dává přednost tzv. „hotovým řešením“, která obsahují kromě vlastního databázového systému i nabídku aplikačního softwaru, který úřad zajímá nejvíce.

Z předchozích odstavců tedy vyplynuly další parciální cíle této diplomové práce, a to hlouběji proniknout do problematiky databázových systémů a ukázat použitelnost databázových systémů v instituci veřejné správy.

K naplnění těchto záměrů, byly stanoveny dílčí cíle.

- Vymezit základní pojmy týkající se problematiky databázových systémů a ohlédnout se do minulosti k základním kamenům databázových systémů.
- Představit databázové systémy dostupné na českém trhu, které jsou relevantní pro úřad veřejné správy.
- Vymezit kritéria, na jejichž základě je možné se rozhodnout pro určitý databázový systém v úřadu veřejné správy a následně představit databázový systém Oracle jako vhodnou volbu.
- Ukázat jako stěžejní bod databázových systému – způsoby zálohování a obnovy dat v konkrétním databázovém prostředí Oracle.
- Ukázat konkrétní způsob zálohy dat a následné obnovy v instituci Magistrátu města Pardubic.

- Navrhnout optimální řešení pro středně velký úřad veřejné správy.

Z uvedených cílu vyplývá struktura práce, která je členěna do pěti stěžejních kapitol.

První kapitola je věnována úvodu. Druhá kapitola se zaměřuje na cíl práce a použitou metodiku, která je základem získávání informací o problematice databázových systémů. Ve třetí kapitole je stručně představena historie databázových systémů, je zde uveden vývoj tohoto „odvětví informatiky“ a představeny základní databázové modely. Rovněž je zde představen i komunikační jazyk SQL, který rovněž prošel vývojem a stal se standardem na poli databázových systémů.

Ve třetí kapitole je rovněž pozornost věnována jednotlivým databázovým platformám, které svými parametry vyhovují databázovým systémů, vhodným pro veřejnou správu a jsou tedy vhodnými kandidáty na nasazení v konkrétných institucích veřejné správy.

Čtvrtá kapitola je již úzce zaměřena na databázový systém Oracle. Je zde popsána architektura databázového systému Oracle, jsou zde vymezeny a vysvětleny základní pojmy související s problematikou Oracle databáze.

Pátou kapitolu lze považovat za stěžejní. Je zaměřena na možnosti zálohování databáze Oracle. Je zde představena většina možností zálohování a zpětné obnovy databáze. Zejména je charakterizován model exportu a importu, zálohování offline a zálohování pomocí nástroje RMAN.

Tato kapitola také představuje výsledky mé praktické práce zkušenosti, kdy jsem aplikoval zálohovací a obnovovací procesy databáze Oracle v konkrétním prostředí instituce veřejné správy – Magistrátu města Pardubic. Výsledkem je konkrétní postup zálohy databáze Oracle na operačním systému HP UNIX True 64 a následná obnova databáze Oracle na operační systém Windows. Mimo velké zálohovací a obnovovací schopnosti je ukázána i nezávislost na operačním systému při kritické obnově dat, kdy se pravděpodobně může stát, že při pádu serveru s operačním systémem HP UNIX True64 již nebude druhý takový server k dispozici. Musí tedy být obnova provedena na server s jiným operačním systémem, v tomto konkrétním případě na operační systém Windows 2003. Proces zálohování i obnovy se podařil bez větších komplikací a potvrdil tak předpoklady, že databázový systém Oracle je velice silný nástroj pro databáze

institucí veřejné správy. Výstupem této kapitoly je použitelný skript pro zálohu i obnovu databáze Oracle, který může být využit prakticky na jakoukoli verzi databáze.

Závěr páté kapitoly obsahuje zobecnění praktických výsledků, kde je navrženo optimálního řešení pro středně velký úřad veřejné správy. Jako jedno z mnoha správných řešení lze navrhnout databázový systém Oracle, který svojí charakteristikou a vlastnostmi je dle mých zkušeností ideální volbou.

Díky vlastní zkušenosti jako implementátora databázové systému Oracle v instituci Magistrátu města Pardubic může být ovlivněno posuzování vhodnosti databázového systému pro středně velký úřad veřejné správy. Parametry, které by databázový systém měl splňovat, jsou uvedeny v předešlých kapitolách a je zde jistota, že rozhodně více platforem těmto kritériím vyhovuje. Pokud se posuzuje databázový systém z hlediska zálohování, stability, spolupráce s ostatními aplikacemi, robustnosti, odolnosti na vznik anomálií, škálovatelnosti a z mnoha dalších hledisek, pak Oracle je ideální volbou. V neprospěch databázového systému Oracle hovoří cenová politika společnosti Oracle, která určuje velmi vysoké ceny nejen za licence používání, ale především za školení administrátorů.

Pokud úřad plánuje nákup databázového systému, jistě by neměl opomenout do svých plánů databázový systém Oracle zahrnout. Vše závisí na zkušenosti IT specialistů, kteří danou platformu vybírají. Jako ideální řešení, pokud se úřad rozhodne investovat do Oraclu, je i investovat do školení administrátora. Pro budoucí databázové administrátory jsou k dispozici tři školení, která jsou odstupňována obtížností správy databáze. První školení je zaměřeno na základní administraci databáze a je vhodné pro první seznámení se s databází Oracle. Druhé školení je zaměřeno na prohloubení administrátorských dovedností. Poslední školení je zaměřeno na optimalizaci a ladění databáze, což je z hlediska administrátora nejužitečnější školení. Co se týče finančních nákladů, tak cena každého školení je 50 000 Kč, tedy dohromady 150 000 Kč, což lze považovat za vysokou investici. Tato investice se však úřadu mnohonásobně navrátí v podobě zabezpečených, řádně zálohovaných dat. Konkurenční společnost Microsoft zvolila poněkud jinou strategii, která vychází pouze ze dvou školení, která se pohybují

přibližně mezi 20 000 – 25 000 Kč. Finanční možnosti úřadu tedy budou důležitým aspektem pro rozhodování o nákupu a provozování databázového systému Oracle.[50]

Data mají v dnešní době obrovskou hodnotu, zvláště pak data institucí veřejné správy a je tedy žádoucí, aby byla kvalitně udržována na předepsané odborné úrovni, což bývá problém. Tento problém je však nikoliv v databázových systémech, ale v jejich obsluze. Dle informací, které vyplývají z diskuzí pracovníků IT, je zřejmé, že si uvědomují slabé místo. Není však v jejich silách, postihnout veškerou problematiku databázových systémů. Potřebné znalosti o databázových systémech jsou nutností. Na tento problém reaguje diplomová práce, která metodicky prohlubuje, s použitím praktických a použitelných příkladů, znalosti o databázích. Tyto znalosti lze považovat za minimální možné pro provoz a správu konkrétního nejrozšířenějšího databázového systému Oracle.

Nejdůležitější a zároveň nejvíce opomíjenou věcí je zálohování a hlavně obnova databázových systémů, což pochopí každý IT specialista v momentě pádu databázového systému.

Řešení, které nabízí tato práce je universální, a v porovnání s většinou odborné literatury je prakticky vyzkoušené a použitelné. Odborná literatura v mnohých případech pouze naznačuje možné cesty, jež však nejsou dotaženy až do konce. Pokud tedy úřad chce poskytovat občanům kvalitní služby, je třeba zajistit chod informačního systému a k tomu je nutné, aby tento úřad zaměstnával specialitu IT, který databázové systémy spravovat umí.

[50] V příloze číslo 2 jsou ukázky certifikací, které je možno v rámci školení získat.

7 SEZNAM LITERATURY

[1] *Extensible Markup Language (XML)* [online]. 2009 [cit. 2009-03-24]. Dostupný z WWW: <http://www.slunecnice.cz/sw/xml/>.

[2] Eykel Xandria: *Administering a Microsoft SQL Server 2000 Database.* [Great Britain?]: Microsoft Corporation, 2000. Material No: 2072ACP.

[3] Fáber, Roman. *Databázové systémy pro GIS* [online]. 2007 , 04.05.2007 [cit. 2009-03-24]. Dostupný z WWW: <http://www.isvs.cz/produkty-a-sluzby/databazove-systemy-pro-gis-iiiii-dil.html>.

[4] Greenwald, Richard. *10 things to know about databases?* [online]. c2009 , March 22 [cit. 2009-03-24]. Text v angličtině. Dostupný z WWW: <http://blogs.oracle.com/RICKG/2009/03/10_things_to_know_about_databa.html>.

[5] Jankovský, Zbyněk. Armagedon: Soumrak databází: Smutné zhodnocení sestupných trendů v databázích. *Connect! : Databáze na tahu*. 2009, roč. XV., č. 2, s. 74.

[6] Klimeš, Jan. Informační systémy s přívlastkem: Business Intelligence. *Informační systémy a systémová integrace* [online]. 2008 [cit. 2009-03-12]. Dostupný z WWW: <http://web.ortex.cz/docs/Co-noveho-v-BI.pdf>.

[7] Klimeš, Lumír. *Slovník cizích slov*. 5. vyd. Praha : Státní pedagogické nakladatelství Praha, 1995. 855 s. číslo publikace: 2-54-13/5b.

[8] Kocan, M. *Svět patří nám.* Professional Computing:Magazín pro IT profesionály, září 2005, roč. VI, str. 38-40.

[9] Kocan, M. *Vyberte si databázi podle gusta.* Connect!:Sítě, komunikace, systémy a bezpečnost, září 2004, roč. IX, str. 12.

[10] Kocan, Marek. *Vývoj databází a jejich trhu.* Connect!. 2009, roč. XV., č. 2, s. 74.

[11] Kocan, Marek. *Databáze dneška potřinácté* [online]. Databázový svět, 2009 , 16.2.2009 [cit. 2009-02-19]. Dostupný z WWW: <http://www.dbsvet.cz/view.php?cisloclanku= 2009021601>.

[12] Lacko Luboslav: *SQL – Hotová řešení.* Brno: Computer Press, 2003. 289 s. ISBN 80-7226-975-5.

[13] Lacko, Luboslav. *Oracle : Správa, programování a použití databázového systému*. [s.l.] : Computer Press, 2007. 576 s. ISBN 978-80-251-1490-2.

[14] Loney Kevin, Theriault Marlene: *Mistrovství v Oracle: Kompletní průvodce tvorbou, správou a údržbou databází.* 1. vydání.
Praha: Computer Press, 2002. 847 s. ISBN 80-7226-635-7.

[15] Oracle Technology Network: *Oracle9i Database Release 2 (9.2) Documentation User, Administrator, and Developer Guides* [online].

[16] *Oracle9i Database Generic Documentation Addendum Release 2 (9.2)*.
Part Number: A97283-01. 2006.
URL: <http://download-uk.oracle.com/docs/html/A97283_01/toc.htm>.

[17] Pokorný, J., HALAŠKA, I. *Databázové systémy*. Praha: ČVUT, 2003.
ISBN 80-01-02789-9.

[18] Pokorný, M.: *Databázový svět* [online]. Vyvíjíme databázový a informační systém IV. 2005 [cit.2009-01-02].
URL: <http://www.dbsvet.cz/view.php?cisloclanku=2004052601>.

[19] Radovan, Banan.cz. *Co je to LDAP* [online]. 2006 , 28.04.2006 [cit. 2008-11-12]. Dostupný z WWW: <http://www.owebu.cz/linux/vypis.php?clanek=830>.

[20] Skřivánek, František. *Databázový svět* [online]. Známe držitele ocenění databázový produkt roku. 2005 [cit.2008-12-02].
URL: <http://www.dbsvet.cz/view.php?cisloclanku=2005120201>.

[21] Skřivánek, František. *Vítejte ve světě SQL* [online]. Databázový svět, 2008, 16.12.2008 [cit. 2009-01-12]. Dostupný z WWW:
<http://www.dbsvet.cz/view.php?cisloclanku=2008121601>.

[22] Sodomka, Petr. *Informační systémy v podnikové praxi*. [s.l.] : Computer Press, 2006. 352 s. ISBN 80-251-1200-4.

[23] Šimonová, S., Panuš, J., Najman, K.: *Databázové systémy II – SQL, přístup k datovým zdrojům*. Určeno pro posluchače kombinovaného studia Fakulty ekonomicko-správní. Pardubice: Univerzita Pardubice, 2006. 100 s.
ISBN 80-7194-845-4.

[24] Naiman, Michal. *Mé jméno je .NET* [online]. 2002 , 01.03.2002 [cit. 2009-01-07]. Dostupný z WWW: <http://interval.cz/clanky/me-jmeno-je-net/>.

8 PŘÍLOHY

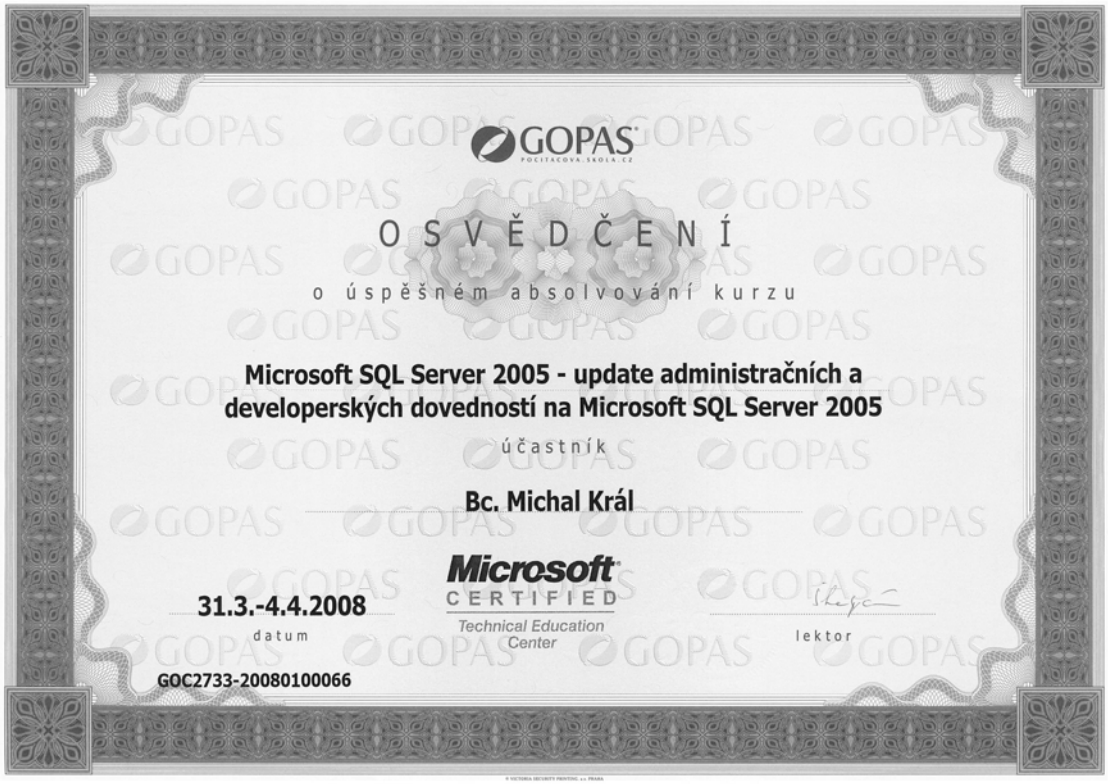
Seznam příloh:

1. Seznam obrázků, tabulek a grafů.
2. Ukázky certifikátů, které je možno v rámci školení databází získat.
3. Seznam zkratek.

Příloha 1

Seznam obrázků, tabulek a grafů

Příloha 2



GOPAS

POČÍTAČOVÁ ŠKOLA .CZ

O S V Ě D Č E N Í

o úspěšném absolvování kurzu

Microsoft SQL Server 2005 - update administračních a developerských dovedností na Microsoft SQL Server 2005

účastník

Bc. Michal Král

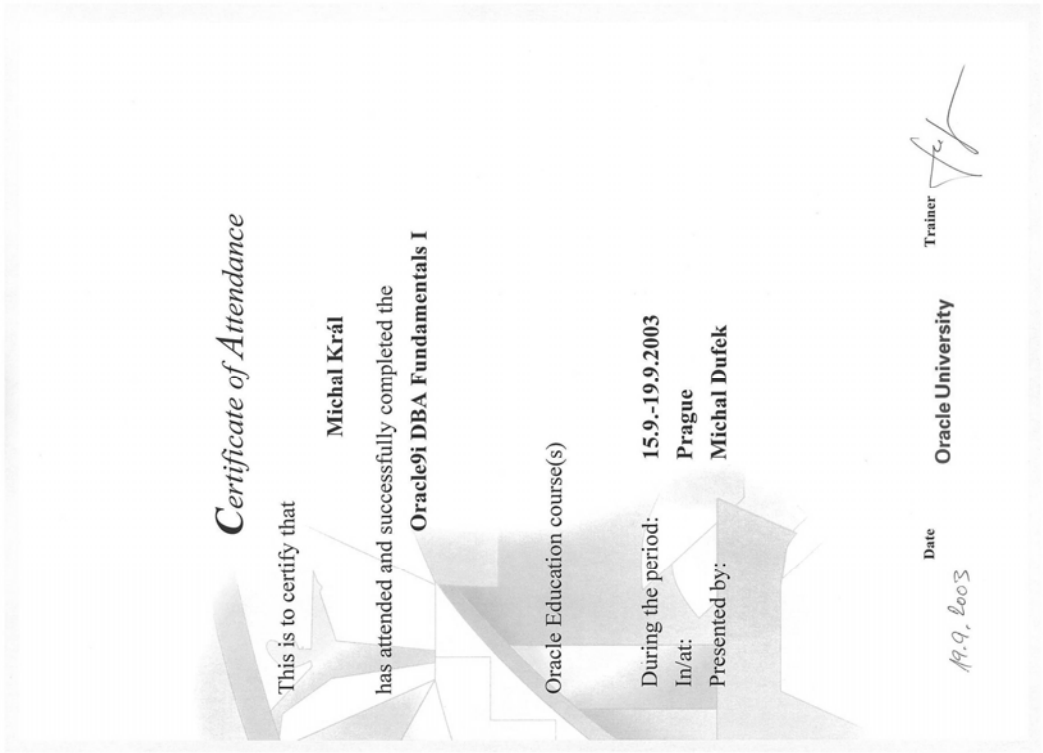
31.3.-4.4.2008

datum

Microsoft CERTIFIED Technical Education Center

lektor

GOC2733-20080100066



Certificate of Attendance

This is to certify that

Michal Král

has attended and successfully completed the

Oracle9i DBA Fundamentals I

Oracle Education course(s)

During the period: **15.9.-19.9.2003**

In/at: **Prague**

Presented by: **Michal Dufek**

Trainer

Date **Oracle University**

19.9. 2003


GOPAS
POCITACOVA.SKOLA.CZ
OSVĚDČENÍ
o úspěšném absolvování kurzu
Oracle9i: ladění výkonnosti serveru
účastník
Michal Král, DiS.
14.-18.6.2004
datum
ORACLE®
lektor
OR-1-O9iR2_PT-20040100003


GOPAS
POCITACOVA.SKOLA.CZ
OSVĚDČENÍ
o úspěšném absolvování kurzu
Oracle9i: Administrace databáze - zálohování a obnova
účastník
Michal Král
5.-9.4.2004
datum
ORACLE®
lektor
OR-1-O9iADM2-2004100002

Microsoft CERTIFIED Partner | Learning Solutions

CERTIFICATE OF ACHIEVEMENT

THIS IS TO CERTIFY THAT

MICHAL KRÁL

HAS SUCCESSFULLY COMPLETED THE FOLLOWING COURSE

ADMINISTERING A MICROSOFT SQL SERVER 2000 DATABASE

CONDUCTED BY MICROSOFT CERTIFIED PARTNER FOR LEARNING SOLUTIONS

OKsystem Ltd.

MCT signature

August 5, 2005
date

Certificate of Attendance

This is to certify that

Michal Král

has attended and successfully completed the

Oracle 9i Database Administration Fundamentals II.

Oracle Education course(s)

During the period: **5.4.-9.4.2004**

In/at: **Prague**

Presented by: **Petr Macák**

9.4.2004

Date

Oracle University

Trainer

Příloha 3

Seznam zkratek

ASE	Adaptive Server Enteprise
API	Application Programming Interface
ASP	Active Server Pages
BI	Business Intelligence
CSP	Caché Server Pages
DBS	Database System
DDL	Data Definition Language
JSP	Java Server Pages
LDAP	Lightweight Directory Access Protocol
MPP	Massively Parallel Processors
NLS	National Language Support
OEM	Original Equipment Manufacture
OLAP	Online Analytical Processing
OLTP	Online Transaction Processing
QBE	Query By Example
RAC	Real Application Clusters
RMAN	Recovery Manager
SGA	System Global Area
SMART	Self Managing And Resource Tuning
SMP	Symmetric Multi Processing
SQL	Structured Query Language
SŘBD	Systém Řízení Báze Dat
SSL	Spisová Služba
TPC	Total Product Cost
XML	Extensible Markup Language