Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Diplomová práce

Zabezpečení databázově koncipovaných evidencí malých a středních podnikatelských subjektů

Bc. Monika Debreczényiová Krammerová

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Bc. Monika Debreczényiová Krammerová

Systémové inženýrství a informatika
Informatika

Název práce

Zabezpečení databázově koncipovaných evidencí malých a středních podnikatelských subjektů

Název anglicky

Assurance of database-based records of small and middle enterprises

---

Cíle práce

Diplomová práce je zaměřena na problematiku zabezpečení provozu databázově koncipovaných informačních zabezpečení malých a středních podnikatelských subjektů. Hlavní náplní této práce je:
a) objasnit teoretické principy relačně databázové technologie v kontextu s problematikou zabezpečení databázových evidencí,
b) zmapovat momentální stav této problematiky a vymezit její relevantnost včetně požadavků na ni kladených,
c) navrhnout přijatelné řešení těchto požadavků,
d) ověřit funkčnost navržených záležitostí,
e) ověřené záležitosti zobecnit pro další možná uplatnění.

Metodika

Použitá metodika zadané diplomové práce bude založena na studiu a analýze dostupných informačních zdrojů a případných existujících řešení v dané oblasti. Práce se bude opírat o metody, techniky a postupy zahrnuté v relačně databázové technologii. Navrhované řešení bude zohledňovat identifikované požadavky a očekávání spojená s řešenou záležitostí. Na podkladě syntézy teoretických poznatků a dosažených výsledků budou formulovány závěry této diplomové práce a následně zobecněny pro další možná použití.

Závazný harmonogram vypracování diplomové práce:

Vymezení teoretických principů řešené problematiky – předmět 1. zápočtu z DP – do 4.9.2019.

Zmapování momentální situace řešené problematiky včetně vymezení s tím souvisejících požadavků – do 30.11.2019.

Navržení konkrétního řešení – do 31.1.2020.

Ověření a zobecnění navrhovaných záležitostí – předmět 2. zápočtu z DP: do 25.3.2020.

Doporučený rozsah práce

55-65 stran

Klíčová slova

Relačně db technologie, zabezpečení evidovaných dat, model kvality dat, práva přístupu, přístupové role, databázový pohled

---

Doporučené zdroje informací

GROFF, James R. a Paul N. WEINBERG. SQL: kompletní průvodce. 2005. Brno: CP Books, 2005. Programování. ISBN 8025103692.

HENDERSON, Ken. Mistrovství v Transact-SQL. Praha: Computer Press, 2000. Profi. ISBN 8072263935.

MORKES, David. Microsoft SQL Server 2000: tvorba, úprava a správa databází. Praha: Grada, 2004. Podrobný průvodce začínajícího uživatele. ISBN 8024707322.

URMAN, Scott, Ron HARDMAN a Michael MCLAUGHLIN. Oracle: programování v PL/SQL. Brno: Computer Press, 2008. Programování (Computer Press). ISBN 978-80-251-1870-2.

VALENTA, M., POKORNÝ, J. Databázové systémy. Praha: České vysoké učení technické v Praze, 2013. ISBN 978-80-01-05212-9.

Předběžný termín obhajoby

2020/21 ZS – PEF (únor 2021)

Vedoucí práce

doc. Dr. Ing. Václav Vostrovský

Garantující pracoviště

Katedra informačního inženýrství

Elektronicky schváleno dne 19. 2. 2020

Ing. Martin Pelikán, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 19. 2. 2020

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 24. 03. 2021

Čestné prohlášení

Prohlašuji, že svou diplomovou práci "Zabezpečení databázově koncipovaných evidencí malých a středních podnikatelských subjektů" jsem vypracovala samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autorka uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 25.03.2021 _____

Poděkování

Ráda bych touto cestou poděkovala doc. Ing. Václavu Vostrovskému, Ph.D., za odborné vedení diplomové práce a za cenné rady týkající se jejího zpracování.

Zabezpečení databázově koncipovaných evidencí malých a středních podnikatelských subjektů

Abstrakt

Hlavním cílem diplomové práce je vytvořit databázový návrh, který je zaměřen na problematiku zabezpečení provozu databázově koncipovaných informačních zabezpečení malých a středních podnikatelských subjektů.

Kromě objasnění teoretických principů relačně databázové technologie je významnou částí definice požadavků na datový model, ovlivněna především současnými trendy v oblasti datové evidence malých a středních podnikatelských subjektů.

Stěžejní částí je ovšem samotný návrh konceptuálního datového modelu odpovídající požadavkům stanovených v předchozí části.

Klíčová slova: relačně databázové technologie, zabezpečení evidovaných dat, práva přístupu, přístupové role.

Assurance of database-based records of small and middle enterprises

Abstract

The main aim of the thesis is to create a database proposal that is focused on the issue of ensuring the operation of database-designed information security of small and medium-sized enterprises.

In addition to explaining the theoretical principles of relational database technology, an important part of the definition of data model requirements is influenced mainly by current trends in data records of small and medium-sized enterprises.

The key part, however, is the design of a conceptual data model that meets the requirements set out in the previous section.

Keywords: security of recorded data, access rights, access roles, relational database technologies

Obsah

Seznam obrázků

Seznam tabulek

1. Úvod

Tématem diplomové práce je návrh řešení problematiky zabezpečení provozu databázové evidence potřebné pro úspěšné fungování malých a středních podnikatelských subjektů. Téma bylo zvoleno z důvodu osobního zájmu o tuto oblast, ale také kvůli možnosti využití teoretických poznatků z České zemědělské univerzity.

Databáze je pojem, který již není nikomu cizí. V této době se data všude shromažďují v databázových systémech. Právě proto se v dnešní době žádná společnost neobejde bez informačních technologií. Může se stát, že společnost nemá na vývoj databáze kvalifikované zaměstnance, proto je tady možnost využít nabízené služby dodavatelských společností, které se zabývají právě touto problematikou. Výběr správné dodavatelské společnosti je klíčovým faktorem pro kvalitní a bezproblémový chod založení databázového systému do běžného života společnosti. V dnešní době si může společnost vybrat z mnoha databázových produktů přímo od firem Oracle, Microsoft, IMB a dalších, které se zabývají právě danou problematikou. Je tady také možnost využít prostředníka jako dodavatelskou společnost, která zpracuje veškeré požadavky a spolupracuje z jednou z výše zmíněných firem.

Pro zavedení nové databáze nebo zlepšení zabezpečení již existující databáze v společnosti je nutný určit přesné znění problému a vymezit na realizaci dostatečné množství prostředků. Dále se musí zpracovat časová os, která je důležitá pro sled událostí, které musí na sebe navazovat, aby se po určitém čase dospělo k cíli.

Důležitou částí této práce je zabezpečení databáze ze strany zaměstnanců firmy. Největším útočníkem v databázi je samotný uživatel. Uživatel musí pochopit svá práva v databázi dále pak to, že jeho jednání v databázi je kontrolovatelný. Toto všechno musí svého zaměstnance naučit právě zaměstnavatel. Bylo by jednoduché dát zaměstnanci práva, z kterých by nevyplývaly také možné postihy. Když je tato otázka vyřešená, může se přejít do samotné práce s databází.

Ještě před samotním udělovaní práv je na prvním místě správně zpracovaná databáze ve formě tabulek. Tyto tabulky musí být zpracovány podle pravidel v relační databázi.

Po vymezení práv a povinností uživatele se můžou zpracovat údaje o uživatelích. Po vzniku uživatelských účtů se zpracuje profil, který omezí a kontroluje přístup uživatele do databáze. Dále se pak zpracují uživatelé do rolí a vytvoří se pohledy pro uživatele, kteří nepracují přímo s databází, jenom potřebují vidět data z ní. V tomto bodě se také společnost musí zamyslet o přístupu uživatele k citlivým datům. Citlivá data jsou pro společnost nevyčíslitelná, proto se k takovým datům musí přístup omezit na maximum.

Po úspěšném zavedení nové databáze do provozu i se zabezpečením, musí být přihlášení uživatele jednoduché, intuitivné a samozřejmě musí být podle předem stanovených pravidel, jak již bylo zmíněno v textu výše. Skladba loginu a hesla je dána a musí umožnit uživateli přihlásit se do databáze a pracovat v rozmezí, které potřebuje k plnění svých pracovních povinností.

Je velice nutné, aby se společnosti zamysleli nad svými daty a věnovaly se zabezpečení databáze. Zabezpečení databáze je konec konců v jejich zájmu a je proto na místě vynaložit na toto zabezpečení finanční zdroje. Tato investice je z dlouhodobého hlediska výhodná investice.

Dané téma je aktuální a tato skutečnost se do budoucnosti nezmění, proto je hlavním důvodem stále pracovat na zabezpečení a kontrole přístupu uživatelů do databáze na místě a nesmí se to podcenit, protože ztráta v podobě ztráty dat a důvěryhodnosti je nevyčíslitelná.

2. Cíl práce a metodika

2.1. Cíl práce

Diplomová práce je zaměřena na problematiku zabezpečení provozu databázově koncipovaných informačních zabezpečení malých a středních podnikatelských subjektů. Hlavním cílem této předkládané diplomové práce je návrh zabezpečení databázové evidence v malém podnikatelském subjektu, které povede ke snížení neoprávněného přístupu vevnitř společnosti, a to konkrétně do samotné databáze, dále omezí přístup uživatelům k samotným datům a práce s nimi. Pro tyto účely bude vytvořená fiktivní databáze s daty v rozsahu dostačující pro zabezpečení přístup a také uživatelé.
Dílčími stěžejními kroky daného zabezpečení budou

1. zmapování současného stavu a zjištění požadavků na nový systém
2. návrh datového modelu
3. požadavky na přístup do databáze
4. realizace zabezpečení na základě daných požadavků
5. testování přístupu do databáze, přístupu k datům a práce s nimi

Těchto pět stěžejních dílčích cílů diplomové práce jsou dále podrobněji rozděleny v kapitole 2.2 Metodika.

2.2. Metodika

Metodika této diplomové práce bude založena na studiu a analýze dostupných informačních zdrojů a existujících řešení. V úvodě diplomové práce bude vypsána teorie, která se bude věnovat základům SQL jazyka a následně bude popsána možnost zabezpečení databáze, v rozsahu pro pochopení problematiky zvolené práce. Stěžejními metodami předkládané práce budou metody relačně databázové technologie, jako jsou datové modelování, datová normalizace, integrita dat a techniky přidělování práv přístupu.

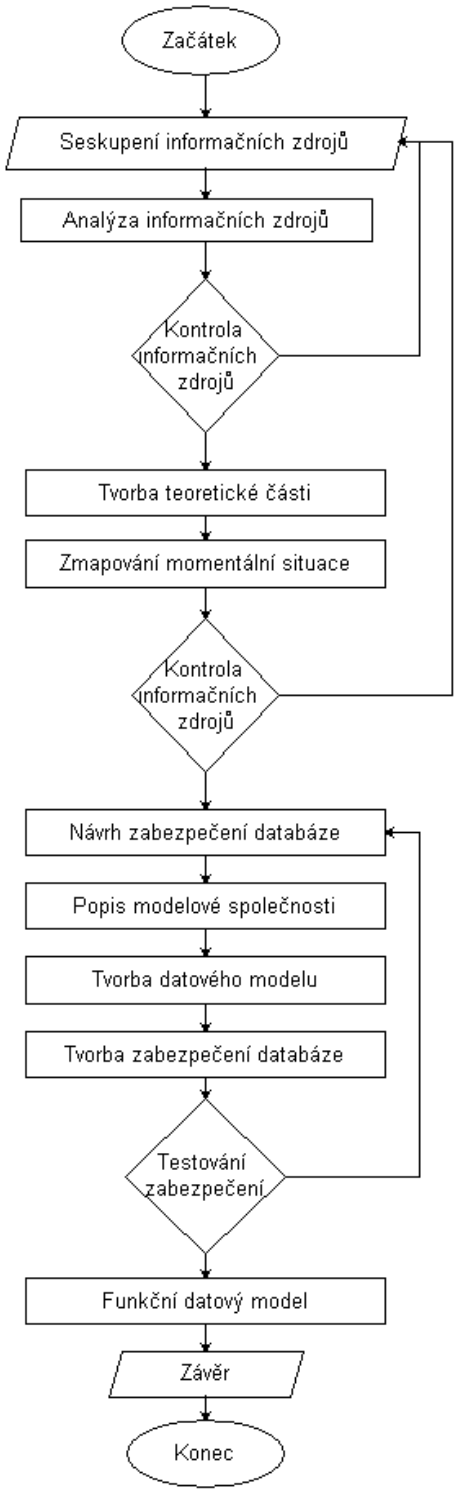
Principy zabezpečení databáze, mezi které patří přehled dat v databázi a možný přístup a práce s nimi. Dále pak nastavení pravidel přístupu do databáze, práce s uživatelskými účty, hesly, tvorba rolí, profilů a pohledů.

Zvolená vlastní metodika znázorněná vývojovým diagramem je následně stručně popsána:

1. Seskupení informačních zdrojů: metodika řešené problematiky je založena na analýze a studiu dostupných informačních zdrojů a existujících řešení v dané oblasti.
2. Podrobná analýza získaných zdrojů: informační zdroje jsou získány z odborné publikace, které se věnují problematice relační databáze a jejímu zabezpečení.
3. Kontrola informačních zdrojů – v případě nutnosti doplnit chybějící informace je potřebné opakovat první bod, v opačném případě se zpracuje teoretická část
4. Tvorba teoretické části: teoretická část je tvořena pomocí odborných informačních zdrojů v dané oblasti, daná část se věnuje tvorbě databáze a jejímu zabezpečení.
5. Zmapování momentální situace řešené problematiky: toto je náročný krok, je to z důvodu toho, že získat přístup do funkční databáze nebude jednoduchý. Zmapování bude muset být na teoretické bázi.
6. V tomto bodě může znova probíhat kontrola informačních zdrojů a v případě nutnosti se chybějící informace doplní
7. Návrh zabezpečení databáze: bude tvořen na základě teoretické části, v které jsou popsány informace z odborné literatury a ty budou využity k navržení zabezpečení databáze v modelové společnosti
8. Popis modelové společnosti, zjištění současného stavu databáze a její zabezpečení: v tomto bodu by se v reálném projektu na zabezpečení databáze museli realizovat střetnutí s uživateli a s vedením společnosti. Tyto kroky budou simulovány pro vznik podrobného popisu společnosti a chodu v ní.
9. Použití vstupních dat získaných při popisu společnosti na tvorbu datového modelu: před samotnou tvorbou modelu budou vznikat tabulky modelové společnosti, které budou následně naplněny daty tak aby bylo možno navrhnout zabezpečení databáze.
10. Tvorba návrhu zabezpečení databáze: toto je bod v kterém se sepíše návrh postupu pro zabezpečení databáze
11. Návrh zabezpečení podle požadavků od společnosti a zároveň od uživatelů: v tomto bodu se budou realizovat navrhnutá řešení pro zabezpečení databáze
12. Testování zabezpečení – testováním zabezpečení databáze se bude zjišťovat funkčnost zabezpečení a v případě nedostatečného zabezpečení se opraví nebo doplní oprávnění.

Testování bude probíhat v sledu postupních kroků, které povedou k zjištění, jestli zabezpečení je úspěšné a vyhovuje modelové společnosti.

13. Po úspěšném testování vzniká funkční datový model
14. Závěr



Obrázek 1 Schématické znázornění postupu řešení – vlastní tvorba

3. Teoretická východiska

3.1. Stručná historie jazyka SQL

Historie jazyka SQL začala v laboratoří společnosti IBM v San Jose v Kalifornii. Zde byl na konci sedmdesátých ket dvacátého století jazyk SQL vyvinut. Zkratka SQL znamená *Structured Query Language* – strukturovaný dotazovací jazyk a samotný jazyk je často označován jako "sequel". Původně byl vyvinut pro produkt společnosti IBM s názvem DB2 což je relační databázový systém neboli RDBMS, který lze i nyní zakoupit pro nejrůznější platformy a prostředí. Ve skutečnosti byla existence relačního databázového systému možná právě díky jazyku SQL. Na rozdíl od do té doby vytvořených procedurálních jazyků nebo jazyků třetí generace jako cobol, nebo C, se jedná o jazyk neprocedurální. [2]

3.2. Podoba dnešní databáze

Počítačová technologie zapříčinila zásadní obrat ve způsobu práce podniků na celém světě. Informace, které byly ukládány ve skladištích plných registrů, mohou být dneska okamžitě nalezeny klepnutím myši. Objednávky zadané zákazníky v jiných zemích je možné ihned zpracovat ve výrobní firmě. [1]

Ačkoli před dvaceti lety bylo mnoho těchto informací přeneseno na podnikové centrální počítačové databáze, v kancelářích se stále pracovalo v neinterakčním prostředí. Pokud měl být proveden dotaz, někdo to oznámil oddělení správy informačních systému a poté byla požadovaná data co nejdříve doručena, ovšem často ne dostatečně brzy. [1]

Kromě vývoje modelu relační databáze zde byly další dvě technologie, které vedly k rychlému růstu toho, čemu dnes říkáme databázové systémy klient-server. První důležitou technologií byl osobní počítač. Ne příliš drahé, snadno použitelné aplikace, umožňovaly zaměstnancům nebo i domácím uživatelům počítačů vytvářet dokumenty a udržovat rychle a náležitě data. Uživatelé si zvykli na neustálé zdokonalování svých systémů, protože rychlost změn byla tak vysoká, jak se snižovala cena pokročilejších systémů. [1]

3.3. Jazyk SQL

Jazyk SQL je nástroj pro organizování, správu a získávání dat uložených v počítačové databázi. Jak vyplývá z názvu, SQL je počítačový jazyk, který se používá pro komunikaci s databází. Ve skutečnosti SQL pracuje s jedním specifickým typem databáze, který nazýváme relační databáze. Pokud se jedná třeba o podnikový počítačový system, může databáze obsahovat data týkající se zásob, výroby, prodeje nebo mezd. Na osobním počítači bude databáze spíše uchovávat informace o domácím rozpočtu, adresář lidí s jejich telefonními čísly nebo data vytažená z velkého výpočetního systému. Počítačový program, který řídí databázi, se nazývá databázový řídicí systém neboli DBŘS. Název strukturovaný dotazovací jazyk je ve skutečnosti nevhodné pojmenování. Tak především, jazyk SQL představuje mnohem více než jen pouhý dotazovací nástroj, i když to byl jeho původní účel a získávání dat patří mezi jeho nejdůležitější funkce. Lze jej taktéž použít k řízení všech funkcí, které databázový systém svým uživatelům poskytuje.
Prostřednictvím jazyka SQL může programátor nebo správce databáze provádět následující operace:

- Definice dat: SQL dovoluje uživateli definovat strukturu a organizaci uložených dat spolu s definicí jejich vzájemných vztahů.
- Získávání dat: SQL umožňuje uživateli nebo aplikačnímu program z database uložená data získávat a používat.
- Manipulace s daty: SQL umožňuje uživateli nebo aplikačnímu program databázi aktualizovat přidáváním nových dat, odstraňovaním starých nebo změnou již drive uložených dat.
- Řízení přístupu: SQL lze používat k omezení schopnosti uživatele data číst, přidávat a modifikovat, čímž je lze před chánit před neautorizovaným přístupem.
- Sdílení dat: SQL se používa ke sladění sdílení dat mezi vice uživateli a k zajištění toho, že se uživatelé mezi sebou navzájem neruší.
- Integrita dat: SQL definuje v databázi omezení integrity, čímž ji chrání před porušením, které může vzniknout kvůli neúplným aktualizacím nebo systémovým selháním. [1,3]

3.3.1. Pravidla pro relační model

Nejpopulárnějším modelem datového úložiště je relační databáze. Jazyk SQL se vyvinul tak, aby sloužil principům relačního modelu databáze. [2]

Relační model navrhnutý Dr. Coddem představoval pokus o zjenodušení struktury databáze. Eliminoval z databáze explicitní struktury rodič-potomek a místo toho reprezentoval v databázi všechna data jako prosté tabulky datových hodnot sestávající se z řádků a sloupců. Relační databáze je databáze, kde se veškerá pro uživatelé viditelná data striktně uspořádají do tabulky datových hodnot a kde veškeré databázové operace pracují s těmito tabulkami.[3]

Chápeme-li navrhování relací (tabulek) jako styl modelování, hovoříme o relačním modelu dat. Základním pojmem a současně i konstrukčním nástrojem v relačním modelu dat je relace. Také v souvislosti s SQL se hovoří o relačním modelu dat a o relačních databázích, je však běžné se setkat s pojmem tabulkový model dat. Relační model dat naplňoval tyto ideje tak charakteristické pro databáze: [4]

- důsledně se oddělují data, která jsou chápána jako relace, od jejich implementace
- při manipulaci s daty se nezajímáme o přístupové mechanismy k datům obsaženým v relacích
- pro manipulaci s daty jsou k dispozici dva silné prostředky – relační kalkul a algebra, které slouží jako základ uživatelských relačních jazyků
- pro omezení redundance dat v relační databázi jsou k dispozici pojmy umožňující normalizovat relace, tj. navrhovat potřebné relační databázové struktury podle přísně definovaných kritérií. [4]

3.3.2. Role jazyka SQL

Jazyk SQL je jazykem relačních databází. Jak uvidíme, použití SQL je širší. Naším cílem je předvést jazyk SQL hlavně z pohledu směrem koncového uživatele a chceme ukázat, jak se v něm konstruují požadavky na databázi. Na rozdíl od procedurálních programovacích jazyků, neprocedurální jazyky, mezi které patří i SQL, popisují, co požadujeme od

databáze a nikoli jak to je třeba provést. SQL je neprocedurální jazyk, a dokonce i více než dotazovací jazyk. Je možné v něm definovat data a provádět aktualizace tak, jak je obvyklé. [4]

Deklarace dat v SQL odpovídá popisu běžné dvojrozměrné tabulky s označenými sloupci. Pro práci se schématy tabulek jsou k dispozici tři příkazy CREATE TABLE, ALTER TABLE, DROP TABLE. Netřeba připomínat, že jde o příkazy, které by měly sloužit hlavně správci dat a pro běžného uživatele databáze by měly být znepřístupněny. Jde o definici dat Data definition language DDL. [4]

Příkazy pro manipulaci s daty v SQL zahrnují podobně jako u každého SŘBD příkazy pro výběr dat a příkazy aktualizační. V tomto případě jde o Data manipulation language DML. Výběr dat reprezentuje jediný příkaz SELECT. Mezi aktualizační příkazy se řadí INSERT, DELETE, UPDATE. [4]

Víceuživatelský přístup přinesl problém zadávání práv uživatelů na přístup k tabulkám v databázi. Obvykle ten, kdo vytváří schémata tabulek, je zároveň vlastníkem odpovídajících tabulek, a dalčí osoba nemá do těchto tabulek přístup. Složka WITH GRANT OPTION dovoluje možnost udílet pomocí příkazu GRANT práva dalším uživatelům kromě těch, kteří jsou uvedení za TO. Komplementární příkaz k příkazu GRANT je REVOKE, pomocí kterého lze práva ručit. Co když ale někdo chce použít definiční příkazy CREATE, DROP a ALTER? Práva na tyto příkazy nemohou být ani přidělena ani zrušena. Příkazy může použít pouze vlastník schématu [4]

Speciální oblastí jazyka SQL je Transaction Control Commands TCC, která obsahuje příkazy pro řízení transakcí. Do této skupiny patří příkazy COMMIT, ROLLBACK, SAVEPOINT [3]

COMMIT

Příkaz COMIT ukládá veškeré změny provedené v průběhu transakce. Zadáním příkazu COMIT před zahájením další transakce ujišťujeme systém, že nedošlo k žádným chybám a nečeká se na dokončení předchozích transakcí.[1]

ROLLBACK

Zatímco probíhá transakce, provádí se obvykle také určitý druh kontroly chyb kvůli zjištění, zda transakce probíhá úspěšně. Transakci můžeme odvolat dokonce i po jejím úspěšném dokončení zadáním příkazu ROLLBACK, který však musíme zadat před příkazem COMIT. Příkaz ROLLBACK je nutno vykonat v rámci transakce. Příkaz říká transakci, aby se vrátila na svůj začátek, jinými slovy databáze se vrací do stavu, v jakém byla na začátku této transakce. [1]

SAVEPOINT

Odvolání transakce se zruší celá transakce. Předpokládejme však, že chceme potvrdit „napůl“ transakci uprostřed příkazu, které obsahuje. Jazyk SQL umožňuje uložit transakci prostřednictvím úložného bodu. Od tohoto okamžiku dále platí, že bude-li zadán příkaz ROLLBACK, transakce se odvolá až ke svému úložnému bodu SAVEPOINT. Všechny příkazy, které byly provedeny až do okamžiku bodu, se uloží. [1]

3.1. Relační vztahy

Relační databáze je databáze, kde se veškerá pro uživatele viditelné data strikně uspořádají jako tabulky datových hodnot a kde veškeré databázové operace pracují s těmito tabulkami. [3]

Klíče

Protože řádky relační tabulky nejsou uspořádány, nemůžeme vybrat určitý řádek podle jeh umístění v tabulce. Neexistuje „první řádek“, „poslední řádek“ nebo „třináctý řádek“ tabulky. Jak tedy můžeme specifikovat specifický řádek? Odpověď jsou klíče. [3]

Primární klíč

V dobře navržené relační databázi má každá tabula určitý sloupec nebo kombinaci sloupců, jejichž hodnoty jednoznačně identifikují každý řádek v tabulce. Tento sloupec označujeme jako primární klíč tabulky. [3]

Cizí klíč

Sloupci, jehož hodnota v jedné tabulce odpovídá primárnímu klíči v jiné tabulce, říkáme cizí klíč. Tak jako můžeme tvořit primární klíč tabulky kombinací sloupců, tak i cizí klíč může být kombinací sloupců. Ve skutečnosti bude cizí klíč vždy vložený klíč, pokud se odkazuje na tabulky se složeným primárním klíčem. Samozřejmě počet sloupců a datové typy sloupců v cizím klíči a primárním klíč musí být identické. Cizí klíče jsou stěžejní část relačního modelu, protože vytváří relace mezi tabulkami v databázi. [3]

3.2. Normalizace databáze

Normalizace je proces, při němž dochází k redukci opakování a nadbytečnosti dat v databázi. Kdybychom měli normalizovat první větu tohoto odstavce, mohli bychom ji formulovat takto: „Normalizace je proces redukce nadbytečních informací v databázi“. [1]

První normální forma 1NF

Tabulka splňuje podmínku první normální formy tehdy, pokud všechny sloupce (atributy) jsou atomické, tedy již dále nedělitelné. Jeden sloupec tak nesmí obsahovat více druhů dat. Matematicky řečeno – každý sloupec musí pro daný záznam obsahovat jen skalární hodnotu. [18]

Druhá normální forma 2NF

Tabulka splňuje podmínku pro zařazené do 2NF tehdy, když splňuje podmínku 1NF a každý sloupec vyjma primárního klíče musí být zcela závislý na celém primárním klíči. Druhá normální forma se proto týká jen těch tabulek, které mají více primárních klíčů. Pokud má totiž tabulka jen jeden primární klíč, je podmínka pro 2NF je splněna automaticky. [18]

Třetí normální forma 3NF

Tabulka je v třetí normální formě tehdy, když je v druhé normální formě a současně neexistují závislosti neklíčových sloupců tabulky. [18]

3.3. Integrita dat

Pro ochranu konzistence a správnosti uložených dat relační systém správy databáze zpravidla zavádí jeden nebo více prostředků pro ochranu integrity dat. Tyto prostředky omezují datové hodnoty, lteré lze vložit do databáze nebo vytvořit při její aktualizaci. V relčních databázích se běžně používá několik druhů prostředků pro ochranu itegrity dat: [3]

Entitní integrita

Primární klíč tabulky musí obsahovat v každém řádku jedinečnou hodnotu, která se liší od hodnot v ostatních řádcích. Například každý řádek tabulky Produkty má unikátní sadu hodnot ve sloupcích ID, což jednoznačně udává produkt uvedený v tomto řádku. Duplicitní hodnoty jsou zakázány, protože by neumožnili databázi odlišit jeden produkt od druhého. V systému správy databáze lze takovou jednoznačnost hodnot vynutit. [3]

Referenční integrita

Referenční integrita představuje záruku konzistence a přesnosti dat v databázi. Referenční integrita prostě znamená, že hodnoty v jednom sloupci nějaké tabulky závisejí na hodnotách jiného sloupce v nějaké jiné tabulce. Kupříkladu k tomu, aby měl nějaký zákazník záznam v tabulce Objednávky, musí nejprve existovat záznam tohoto zákazníka v tabulce Zákazník. Integritní omezení dokážou omezením rozsahu hodnot pro daný sloupec také kontrolovat zadávané hodnoty. Integritní omezení by se mělo vytvářet při tvorbě tabulky. Referenční integrita je obvykle řízená prostřednictvím primárních a cizích klíčů. [2]

Doménová integrita

Každý sloupec v databázi má doménu, což je množina povolených hodnot pro tento sloupec. Vzorová databáze používá čísla objednávek, která začíná například 100001, takže sloupec Cilso_Obj je celým kladným číslem větším než 100000. Podobně musí čísla zaměstnanců ve sloupci Cislo_Zam spadat do číselného rozsahu například 001 až 999. V DBŘS lze všechny ostatní hodnoty pro daný sloupec zakázat.[3]

3.4. Budování zabezpečení

Svět je stále menší a informace jsou každým dnem více přístupné. Počítače se výrazně změnily a umožnily zlepšit náš každodenní život. Umožňují nám dělat úžasné věci, lidé mohou sdílet myšlenky na vzdálenosti tisíce kilometrů. V době, kdy je snadné sdílet data, vzrůstá potřeba zabezpečit, aby k některým datům byl omezený přístup. [10]

Představme si například záznamy o našem zdravotním stavu nebo financích. Co by se stalo, kdyby tyto záznamy padly do rukou nepovolaných osob? Následky by mohly být zničující, a to nejen pro nás, ale také pro společnosti zodpovědné za ochranu těchto záznamů. Zabezpečení dat můžeme definovat jako sadu postupů a činností, které společnosti používají, aby uchránily informace před nepovolanými osobami.
Myslíme si, že abychom mohli rozumět budoucnosti, musíme prozkoumat a porozumět minulosti. Proto zde ukážu krátký přehled historie zabezpečení dat. [10]

3.4.1. Vývoj zabezpečení

Jeden z prvních zaznamenaných případů, kdy bylo použito zabezpečení dat se stal už více než před dvěma tisíci lety. Julius Caesar používal primitívní způsob šifrování při zaslání zpráv svým generálům do celého světa. Dnes je tento algoritmus známy jako Caesarova šifra. Význam této zprávy je skryt tím, že se každé písmeno zprávy posune o tři písmena vpravo. Jak mohl Caesar používat takto slabý algoritmus? Jeho šifrování uspělo z následujících důvodů:

- jeho nepřátelé nevěděli nic o šifrování
- tajemství algoritmu bylo důkladně střeženo
- nebyly ještě známy matematické způsoby luštění šifer [10]

Tento algoritmus měl však mnoho problémů a časem by neobstál. Největší problém samozřejmě spočívá v tom, že jakmile by bylo tajemství odhaleno, bylo by snadné číst všechny zprávy.

Dnešní zabezpečení počítačů je založeno na stejných principech, jako kdysi Caesarova šifra. Používají se sofistikované způsoby, kterým se změní vzhled, takže lidé nemohou naši informaci snadno přečíst. Šifrování je změna informace, kterou provádíme proto, abychom tuto informaci skryli. Šifrování nám umožní poslat zprávu, které bude rozumět jenom příjemce, a současně zabrání ostatním lidem, kteří by mohli tuto zprávu číst, aby jí rozuměli také. Klasickým příkladem použití této technologie k utajení informace je šifrování elektronických zpráv posílaných přes internet. Když potom vezmete zašifrované slovo, větu nebo zprávu a aplikujete algoritmus, kterým přeložíte objekt zpět do jeho původní podoby, zprávu rozšifrujete. [10]

3.4.2. Poznejte útočníky

Termínem útočník se budou označovat ty lidi, před kterými chceme chránit naše systémy. Útočníci mohou způsobit škody náhodou či omylem nebo úmyslně. Mezi škody, které útočník může způsobit, patří:

- Smazání našich dat
- Změna našich dat, kterou nedokážeme zjistit
- Přečtení našich dat, které můžou ohrozit pozici naší organizaci
- Zničení našeho systému

Typ útočníků, kterých je třeba se bát, závisí na datech, které chráníme. Je-li chráněna informace veřejná, nemusíme řešit, kdo má mít možnosti ji číst. V některých systémech zase uživatelé mohou modifikovat pouze svá vlastní data. Možné ohrožení i jak mu čelit je nutné vyvodit z naší konkrétní situace. [10]

Vojenské organizace mají různé úrovně data a chrání je před různými typy útočníků. Ve vojenských organizacích a organizacích státní správy lze data klasifikovat od přísně tajných po veřejné. Při ochraně relativně nedůležitých dat můžou být větší hrozbou zvědaví lidé. Organizace nemají vždy dost prostředků nezbytných k zabezpečení každé jednotlivé informace. Vždy tu bude šedá oblast, v níž leží méně důležitá data a která není dostatečně zabezpečená. Je samozřejmé, že roztřídění informací podle důležitosti, a tedy i náležité ochrany záleží na nás. [10]

Všichni důvěřujeme bankám, že uchovávají naše peníze a zpracují naše transakce správně. Největším postrachem a hrozbou pro zabezpečení v bankách je někdo, kdo se snaží získat naše peníze. Uvědomujeme si vůbec, že hrozba zlodějů existuje jak zvenčí, tak zevnitř? [10]

3.4.2.1. Útočníci zevnitř

Tito lidé obvykle mají přímý přístup do sítě a více či méně jim důvěřujeme. Mohou to být zaměstnanci, konzultanti, zaměstnanci na výpomoc nebo snad zvědové v převleku zaměstnanců. A jsou zde také nevinní zaměstnanci, který, ať už z pohodlnosti, nutnosti nebo lenosti, porušili psané zásady zabezpečení nebo se chovají v oblasti bezpečnosti špatně. Nemají žádné zákeřné úmysly, ale přesto jsou nebezpečnou hrozbou.

Zaměstnanci mohou z pohodlnosti sdílet heslo a tím zvyšují riziko, že náš systém bude ohrožen. Jinými slovy, jestliže dokážeme najít důvod, pro který naši uživatelé nedodržují zásady bezpečnosti, nejdete možná řešení, kterým omezíte snahu uživatelů porušovat pravidla. [10]

Hrozby, které představují administrátoři

Největší ohrožení, se kterým je třeba se vypořádat, pochází od administrátorů. Jak mohou být hrozbou právě administrátoři? Už tím, že do nich vkladáme plnou důvěru. Problém je ve střádání zamestnání – náš skvělý administrátor může totiž už zítra pracovat pro našeho největšího konkurenta. Není snadné zabránit někomu, kdo má administrátorská práva, aby jich nezneužil, ale lze stanovit postupy, které mohou jednotlivci ztížit provedení něčeho škodlivého, aniž by byl odhalen. [10]

Hrozby ze strany koncových uživatelů

Koncoví uživatelé představují několik typů obvykle nezáludých hrozeb. Nejlepší způsob, jak vyřešit problémy zabezpečení, které se tkají koncových uživatelů, je kombinace technických řešení a vzdělání. Koncoví uživatelé mají obvykle přidělená data, která mají oprávnění prohlížet, modifikovat, doplňovat a případně i odstranit. [10]

Vliv velikosti organizace

Malé organizece, pokud jde o zabezpečení počítačů, dávajú většinou svým zaměstnancům plnou důveru. Ve větších společnostech tento přístup, kdy zaměstnancům veřím, dokud se neprokáže, že nejsou důvěryhodní, uplatňovat nelze. Při plánovaní zabezpečení je třeba být trochu paranoidní. Neexistuje přesný návod, jak stanovit riziko každé z hrozeb, ale měli bysme vzít do úvahy následující:

- Počet uživatelů interní sítě
- Počet lidí, kteří mají přístup k systému
- Rezdělnení pravomocí uvnitř organizace
- Povahu dat, která chráníme [10]

3.4.2.2. Hrozby přicházející zvenku

Útočníci zvenku většinou nemají informace o našem systému. Proto pečlivý útočník nejprve začne sbírat informace. Nejprve možná začne sledovat, co dělají ostatní uživatelé na síti. Nebo může sbírat informace o verzích programového vybavení a nainstalovaných opravách. Na základě těchto informací potom vetřelec vytvoří plán útoku. Jako například trojské koně. Je ho těžké objevit, protože se v tichosti naistaluje do systému, začne komunikovat s útočníkem a pošle mu informace o našem systému. Tím umožní útočníkovi vstoupit do systému znovu, s vyšším oprávněním. Tato forma útoku byla použita k napadení firmy Microsoft v říjnu roku 2000. Neopatrný zaměstnanec dostal e-mail s přílohou. Když tento pracovník nerozumně otevřel přílohu, vir se naistaloval do systému firmy a komunikoval s útočníkem. Chyba jediného uživatele otevřela virtuální dveře a umožnila útočníkovi dostat se za hranice firmy. Škody naštěstí nebyly vážné, ale tu vidíme, jak díky slabému článku došlo k proniknutí do zabezpečení organizace, navíc organizace, která se zabezpečením velmi vážně zabývá. [10]

3.5. Zabezpečení komponent mimo databázi

Všechny teorie uvedené v následujících kapitolách jsou k ničemu v případě, kdy není zabezpečen přístup k operačnímu systému nebo k vlastnímu fyzickému hardwaru. V této čísti jsou informace o jednotlivých nedatabázových komponentách, které je nutné zabezpečit ještě předtím, než je možné vlastní databázi považovat za bezpečnou. [12]

- **Zabezpečení operačního systému –** pokud databáze Oracle neběží na vlastním vyhrazeném hardwaru, kde jsou aktivované pouze uživatelské účty root a oracle je vhodné provést zabezpečení operačního systému. Vlastní software by měl být nainstalován pod účtem uživatele oracle a nikoli root. Vlastníkem spustitelných souborů softwaru i databázových souborů by měl být jiný uživatel než oracle, čímž se případným hackerům výrazně znesnadní jejich práce. [12]

- **Zabezpečení zálohových médií –** k zálohovacím médiím by měl mít přístup pouze omezený kruh lidí. Zabezpečený operační systém a silná, zašifrovaná hesla v databázi jsou na nic v případě, kdy hacker záložní získá kopie databáze, které může nahrát na vlastní server. To platní o všech serverech, které obsahují replikovaná data databáze. [12]

- **Interní kontroly – kontrola** jednotlivých zaměstnanců, kteří pracují s citlivými databázovým daty jako jsou správce databáze, auditor, správce operačního systému, je naprostou nutností. [12]

- **Osvěta v oblasti zabezpečení –** důležité je, aby databázoví uživatelé pochopili zásady zabezpečení a využívání databáze v prostředí IZ infrastruktury. Informovanost jednotlivých uživatelů by měla odpovídat důležitosti a hodnotě databázových dat pro organizaci. [12]

- **Omezený přístup k hardwaru –** veškerý hardware, na kterém běží databáze, by měl být uložen na zabezpečeném místě, které je přístupné pouze vybrané skupině lidí s využitím bezpečnostního kódu nebo čipové karty. [12]

3.6. Zabezpečení databáze

Daná kapitola se bude zabývat bezpečností databáze. Zejména se bude věnovat různým příkazům a konstrukcím jazyka SQL, které nám umožní spravovat a efektivně řídit relační databázi. Bezpečnost je často opomíjeným aspektem návrhu databáze.
Většina počítačových profesionálů vstupuje do počítačového světa s určitými znalostmi programování nebo hardwaru a často mají sklon soustředit se pouze na tyto oblasti. Jedním z častých problémů je, že vývojáři nezváží všechny aspekty ve vlastní provozní fázi aplikace. Co se stane, když bude aplikaci používat spousta uživatelů v rozlehlé síti?
Naštěstí pro nás výrobci softwaru poskytují většinu nástrojů, které k ošetření tohoto bezpečnostního problému potřebujeme. [1]

Zabezpečení databáze je v každé společnosti nutné a je jedno jestli jde o menší nebo větší společnost. Je důležité aby byla databáze zabezpečená, protože hlavní aktivem společnosti jsou data. Takže je velice důležité aby přístup k těmto datům byl omezený.
To znamená, že ne každý zaměstnanec musí mýt k datům jako jsou platy zaměstnanů přístup. Každý užívatel, který má přístup do databáze musí mít své heslo a jasne stanovenou roli a práva. Není možné aby každý užívatel databáze mohl jak data vkládat, tak upravovať, měnit nebo mazat. Je velice pravdepodobné, že velká větčina uživatelů bude mít právo jen na čtení nikoliv na změny. [3]

Požadavků na zabezpečení typické provozní databáze je mnoho a mění se:

- Data v jakékoli dané tabulce by měla být dostupná některým uživatelům, ale přístup jiných uživatelů by měl být omezen
- Některým uživatelům by mělo být povoleno aktualizovat data v konkrétní tabulce. Ostatním by mělo být povoleno pouze načítání dat
- U některých tabulek by měl být přístup omezen podle sloupců
- Některým uživatelům by měl být zakázán interaktivní přístup pomocí SQL k tabulce, ale mělo by jim být umožněno používat aplikační programy, jež tabulku aktualizují [3]

Jazyk SQL definuje celkový systém zabezpečení databáze a příkazy SQL používané pro bezpečnostní omezení. Schéma zabezpečení v SQL je založeno na třech centrálních prvcích: [3]

- **Uživatelé** – aktéři v databázi. Pokaždé, když DBŘS načítá, vkládá, odstraňuje nebo aktualizuje data, činí tak jménem nějakého uživatele. Systém umožňuje nebo zamezuje akci v závislosti na tom, který uživatel požadavek vytváří. [3]

- **Databázové objekty** – položky, na které lze zabezpečení aplikovat. Zabezpečení se obvykle vztahuje na tabulky a pohledy, ale i na jiné objekty. Chráněny mohou být také celé databáze. Většina uživatelů bude mít práva k používání určitých databázových objektů a k používání ostatních ne. [3]

- **Práva** – akce, které uživatel může provádět na daném databázovém objektu. Uživatel například může mít právo načítat nebo vkládat řádky do konkrétní tabulky, ale nemusí mít právo odstraňovat nebo aktualizovat. Různí uživatelé mohou mít různé sady práv. [3]

3.6.1. Vytváření uživatelů

Pro získání přístupu k databázi a tím i prostředkům svázaným s uživatelským účtem musí uživatel zadat uživatelské jméno. Každé uživatelské jméno má heslo a je mu přiděleno jediné databázové schéma. Některé účty nemusejí mát ve svém schématu žádné objekty, ale místo toho mohou mít přidělena oprávnění pro přístup k objektům v jiných schématech. [12]

Tato část se věnuje informacím o vytváření, aktualizaci a odstraňování uživatelů včetně příkladů syntaxe.

Syntaxe příkazu **create user** je poměrně jednoduchá. Příkaz má několik parametrů, jejichž popis nalezneme v tabulce.

Tabulka 1 Parametry příkazu CREATE USER zdroj [12]

Parametry příkazu CREATE USER	
Uživatelské jméno	Název schématu a vytvářeného uživatele. Uživatelské jméno může mít maximálně 30 znaků a nemůže to být klíčové slovo.
IDENTIFIED {BY heslo | EXTERNALLY | GLOBALLY AS 'externí název'}	Udává způsob autentizace uživatele. Buď to může být databázová autentizace heslem, nebo autentizace prostředky operačního systému, nebo autentizace službou
DEFAULT TABLESPACE Tabulkový prostor	Tabulkový prostor, kde jsou vytvářeny trvalé objekty uživatele. Tento tabulkový prostor může být explicitně specifikován v průběhu vytváření uživatele
TEMPORARY TABLESPACE Tabulkový prostor	Tabulkový prostor, ve kterém jsou vytvářeny dočasné segmenty při provádění operací řazení, vytváření indexů atd.
QUOTA {velikost | UNLIMITED} ON tabulkový prostor	Velikost místa pro objekty vytvářené v daném tabulkovém prostoru. Velikost je v kb nebo Mb
PROFILE profil	Profil přiřazen uživateli. Informace o profilech nalezneme v další části. Pokud není profil specifikován, je použit výchozí profil DEFAULT
PASSWORD EXPIRE	Při prvním přihlášení musí uživatel změnit heslo
ACCOUNT {LOCK | UNLLOCK}	Udává, jestli uživatelský účet bude po vytvoření odemčen nebo uzamčen. Výchozí stav je odemčen.

Je-li zvolena možnost BY heslo, pak systém vyzve uživatele při jeho každém přihlášení k zadání hesla. Jako příklad vytvořím uživatelské jméno:

SQL> CREATE USER Dora INDETIFIED BY pokus;
User created.

Při každém přihlášení s uživatelským jménem Dora bude uživatel vyzván k zadání hesla pokus. Pokud by se zvolila možnost EXTERNALLY, spolehne se databázový systém na přihlašovací jméno a heslo použito při přihlášení do systému. Přihlášením do systému se tedy v podstatě přihlásí uživatel také do databáze. [2]

Heslo lze změnit příkazem **alter user.**
SQL> ALTER USER Dora INDETIFIED BY avokado;
Nyní již Dora nemá heslo „pokus“ ale „avokado“. Dokud ale nebude mít systémové oprávnění CREATE SESSION, nebude se moci ke svému účtu přihlásit:

SQL> grant CREATE SESSION to Dora; [8]

3.6.2. Správa hesel a profilů

Hesla mohou pozbýt platnosti a účty lze po opakovaných neúspěšných pokusech o přihlášení zablokovat. Historii hesel můžeme při provádění změn uchovávat a předejít tak použití dřívějších hesel. Charakteristiky vypršení platnosti hesla jsou u účtu určeny profilem, který je mu přiřazen. Profily vytvořené příkazem **create profile** spravuje správce databáze. [8]

Profily je možné kromě jiného využít také jako autorizační mechanizmus pro vytváření, znovu používání a ověřování platnosti uživatelských hesel. Pomocí těchto pravidel je například možné zajistit, aby heslo mělo nějakou minimální délku a aby sa například v helse objevilo alespoň jedno velké a jedno malé písmeno. [12]
Příkaz CREATE PROFILE slouží ke dvěma účelům. Například je možné vytvořit také profil, který omezuje dobu připojení uživatele na 120 minut.
SQL> CREATE PROFILE lim_connect limit
connect_time 120;

Podobně se dá omezit počet neúspěšných pokusů o přihlášení před uzamčením účtu:
SQL> CREATE PROFILE lim_fail_login limit
fail_login limit failed_login_attempts 8;

Reakce databáze na překročení některého limitu je závislá na typu omezení. Pokud je překročen limit po dobu připojení nebo limit doby nečinosti uživatele, jsou aktivní transakce ukončeny operací ROLLBACK a relace uživatele je odpojena. V této částí se budu věnovat informacím o správě hesel a prostředků s využitím profilů. [12]

Tabulka 2 Parametry hesel profilu. Zdroj [12**]**

Parametry hesel profilu	
FAILED_LOGIN_ATTEMPTS	Počet neúspěšných pokusů o přihlášení před uzamčením účtu
PASSWORD_LIFE_TIME	Doba (ve dnech), po kterou je možné heslo používat. Pokud není určeno jinak parametrem PASSWORD_GRACE_TIME, je nutné heslo po uplynutí této doby změnit.
PASSWORD_REUSE_TIME	Doba (ve dnech), po kterou musí uživatel čekat před tím, než je možné opakovaně využít staré heslo. Používá se současně s parametrem PASSWORD_REUSE_MAX
PASSWORD_REUSE_MAX	Počet změn hesel, který musí nastat předtím, než je možné opakovaně využít staré heslo. Používá se současně s parametrem PASSWORD_REUSE_TIME
PASSWORD_LOCK_TIME	Doba (ve dnech), po kterou je účet po dosažení počtu neúspěšných pokusů o přihlášení, daného parametrem FAILED_LOGIN_ATTEMPTS, uzamčen. Po uplynutí této doby je účet automaticky odemčen.
PASSWORD_GRACE_TIME	Doba (ve dnech), po kterou je možné změnit heslo, které přestalo platit. Pokud v tomto časové intervalu nedojde ke změně hesla, je účet označen jako neplatný a před následujícím úspěšným přihlášením je nutné změnit heslo

	tohoto účtu.
PASSWORD_VERIFY_FUNKCION	Skript jazyka PL/SQL, který obsahuje dodatečný algoritmus kontroly hesla. Pokud je zadána výchozí hodnota NULL, kontrola hesla se neprovádí.

Hodnota parametru **unlimited** udává, že pro využití daného prostředku neexistuje žádné omezení. Hodnota **default** znamená, že parametr přebírá hodnotu z profilu DEFAULT. Parametry **password_reuse_time** a **password_reuse_max** je nutné použít současně. Nastavením jednoho bez druhého nemá žádný uživatelný efekt.
V následujícím příkladu je vytvořen profil, který nastaví hodnotu parametru **password_reuse_time** na 20 dní a hodnotu parametru **password_reuse_max** na hodnotu 5 dní.

```
SQL> CREATE PROFIL lim_reuse_pass limit
        password_reuse_time 20
        password_reuse_max 5;
```

Uživatelé tohoto profilu mohou staré heslo opakovaně použít poprvé po uplynutí 20 dnů za předpokladu, že v této době bylo heslo změněno alespoň pětkrát. Pokud je zadaná hodnota některého parametru, přičemž hodnota druhého parametru je UNLIMITED, uživatel již jednou použité heslo nemůže použít opakovaně. [12]

3.6.3. Vytváření rolí

Kromě výchozích rolí, můžete v Oraclu vytvářet i své vlastní role. Námi vytvořené role mohou sestávat z tabulkových nebo systémových oprávnění či z kombinace obou. [8] V této částí se budeme věnovat informacím o vytváření a správy rolí.

Ve společnosti je mnoho různých skupin a každý člověk dělá pro společnost jednu nebo několik činností.

Každý zaměstnanec má svůj úkol a potřebuje jedno nebo více oprávnění k jednomu nebo několika objektům v databázi. Například ochodníci potřebují pracovat se seznamem zákazníků a s informacemi.

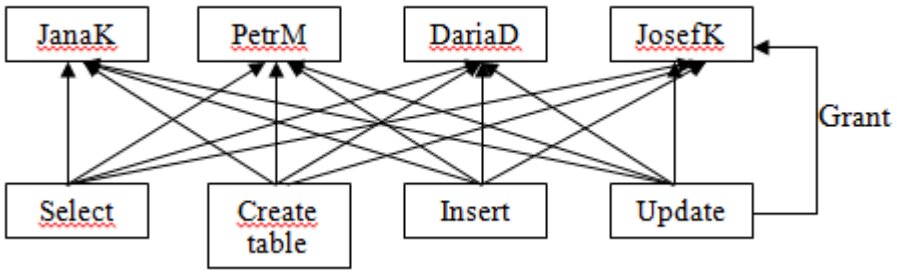
Když obchodník uzavře obchod, informace o obchodu musí zaměstnanec vložit do databáze, aby byl zákazník správně spoplatněn a obchodník dostal svoji provizi.

Každý zaměstnanec, vyžaduje určitou úroveň přístupu k databázi, aby mohl provádět svoji práci. Můžeme vytvořit roli které přidáme oprávnění jako je select k tabulkám, se kterými zaměstnanec potřebuje pracovat.

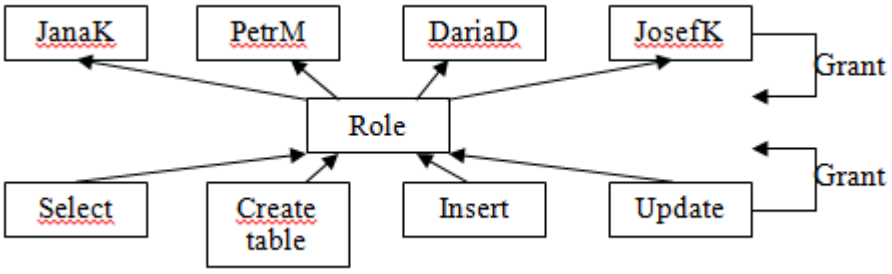
Nebo se může vytvořit role, která bude mít vyšší oprávnění jako je vkládat informace i modifikovat.

Pokud bude společnost velká a úspěšná, bude mít velký počet zaměstnanců, kteří budou uvedené činnosti provádět. Role jsou efektivní prostředek, jak administrovat potřeby všech zaměstnanců a zajistit, že sady oprávnění v databázi jsou konzistentní. [8]

Role je pojmenovaná skupina oprávnění, buď systémových, nebo objektových, nebo kombinace obou, která usnadňuje správu oprávnění. Místo individuálního přidělovaní jednotlivých systémových nebo objektových oprávnění každému uživateli je možné přidělit skupinu systémových nebo objektových oprávnění roli a pak přidělit tuto roli příslušným uživatelům. To výrazným způsobem snižuje vytížení správce databáze, způsobené správou oprávnění uživatelů. Obrázky 2 a 3 znázorňují, jak použití rolí snižuje počet prováděných příkazu **grant** a **revoke.** [12]



Obrázek 2 Přidělování oprávnění bez využití rolí – vlastní tvorba.



Obrázek 3 Přidělování oprávnění s využitím rolí – vlastní tvorba

Pro vytvoření role se použije příkaz **crete role** a pro vlastní vytvoření role je nutné systémové oprávnění CREATE ROLE. Toto oprávnění je obvykle přiděleno pouze správcům databáze nebo správcům aplikace. [12]

SQL> CREATE ROLE pokus NOT IDENTIFIED;
Role created.

Při aktivaci nebo použití přiřazené role se implicitně nepoužívá ani heslo, ani jiný způsob autentizace, klauzule **not identified** je proto nepovinná.

Odstranění role je stejně jednoduché jako vytvoření.

SQL> DROP ROLE pokus;
Role dropped.

Za učelem zvýšení úrovně zabezpečení databáze může správce databáze přidělit rolím hesla. Heslo je roli přiděleno při jejím vytváření. [12]

SQL> CREATE ROLE pokus BY mnbvc125;
Role Create.

3.6.4. Práva

Nyní se budou popisovat práva uživatele v databázi. Každý uživatel v databázi může mít různá oprávnění. Je to dáno pracovním zařazením, to znamená, že správce databáze bude mít určitě jiná práva než uživatel, který potřebuje ke své práci zobrazit data týkajících se zboží.

Existují čtyři základní práva pro práci s tabulkami a pohledy. Jsou to SELECT, UPDATE, INSERT, DELETE.

- SELECT – načítá data z tabulky nebo pohledu
- INSERT – vkládá nové řádky do tabulky nebo pohledu
- DELETE – odstraňuje řádky dat z tabulky nebo pohledu
- UPDATE – upravuje řádky dat v tabulce nebo pohledu

Když vytvoříme pohled pomocí příkazu CREATE VIEW, staneme se vlastníkem pohledu, ale nemusíme k němu nutně obdržet plná práva. Pro úspěšné vytvoření pohledu musíme mít právo SELECT. Jiné to je pro vytvoření tabulky. Tabulka se vytváří pomocí příkazu CREATE TABLE a po jejím vytvoření se stáváme jejím vlastníkem s plnými právy. Ostatní uživatelé k nově vytvořené tabulce nemájí práva ale jako vlastník jim práva můžeme přidelit pomocí příkazu GRANT. [3]

3.6.4.1. Přidělování práv

Základní příkaz GRANT se používá pro přidělování bezpečnostních práv k objektům databáze pro uživatele. Příkaz GRANT zpravidla používá vlastník tabulky nebo pohledu pro přidělení přístupu k datům jiným uživatelům. Příkaz GRANT poskytuje dva zkrácené povely, které můžeme použít při přidělování mnoha práv nebo při jejich přidělování mnoha uživatelům. Namísto uvedení všech práv dostupných pro konkrétní objekt můžeme použít klíčová slova ALL PRIVILEGES. Tento příkaz dává uživateli plný přístup k tabulce. 0bčas můžeme chtít umožnit jiným uživatelům přidělovat práva k objektu, který vlastníme. Toto nám umožňuje WITH GRANT OPTION. [3]

3.6.4.2. Odebírání práv

Příkazem REVOKE lze odebrat některá nebo všechna práva, jež jsme dříve uživateli přidělili. Pomocí příkazu REVOKE můžeme odebírat pouze ta práva, která jsme my dříve přidělili jinému uživateli. Může se stát, že uživatel má také práva, která byla přidělena jinými uživateli. Tato práva nejsou naším příkazem REVOKE dotčena.[3]

Proto je přidělovaní práv nutno podrobně přemyslet a neumožňovat jiným uživatelům přidělovat práva. Pak se nemusíme zaobírat popsaným problémem při odebíraní práv.

Syntaxe příznaků GRANT a REVOKE

GRANT {ALL|<seznam-oprávnění> [(sloupec [sloupec...])]} ON <jméno-tabulky | jméno-náhledu> TO {PUBLIC |<seznam-uživatelů>} [WITH GRANT OPTION]

REVOKE [GRANT OPTION FOR] {ALL|<seznam-oprávnění> [(sloupec [sloupec...])} ON <jméno-tabulky | jméno-náhledu> FROM {PUBLIC |<seznam-uživatelů>}

Tabulka 3 Oprávnění. Zdroj [12]

Oprávnění	Popis
ALTER	Umožní provádět změny tabulky nebo definice sekvence
DELETE	Odstranění řádků z tabulky nebo pohledu
INDEX	Umožní vytváření indexů nad tabulkou
INSERT	Vkládání řádků do tabulky a pohledu
REFERENCES	Umožní vytvoření cizího klíče, který odkazuje na primární nebo jedinečný klíč jiné tabulky.
SELECT	Umožní čtení řádků tabulky a pohledu
UNDER	Umožní vytvoření pohledu založeného na aktuálním pohledu
UPDATE	Umožní aktualizování hodnot v řádcích tabulky a pohledu

3.6.5. Použití pohledů

Pohledy fungují podobně jako objektiv mikroskopu. Je to maska, kterou položíme přes tabulku nebo několik tabulek, a tak změníme obraz, který uživatel vidí. Pohled může obsahovat jen podmnožinu sloupců tabulky nebo ho můžeme napsat tak, aby omezil počet záznamů nebo typ dat, která dotaz vrátí. K čemu jsou dobré pohledy? Když se často spouští komplikovaný dotaz, je jednodušší zadat jednoduchý dotaz na pohled než stále přepisovat složitý dotaz. Když se dotaz změní, je potřeba pouze změnit definici pohledu, a neměnit dotaz ve všech programech. [10]

Pomocí klauzule WHERE můžeme řídit přístup k datům, omezit přístup uživatelů k určitým záznamům nebo sloupcům. Z hlediska bezpečnosti je pohled velice cenný nástroj pro omezení dostupnosti některých dat. [10]

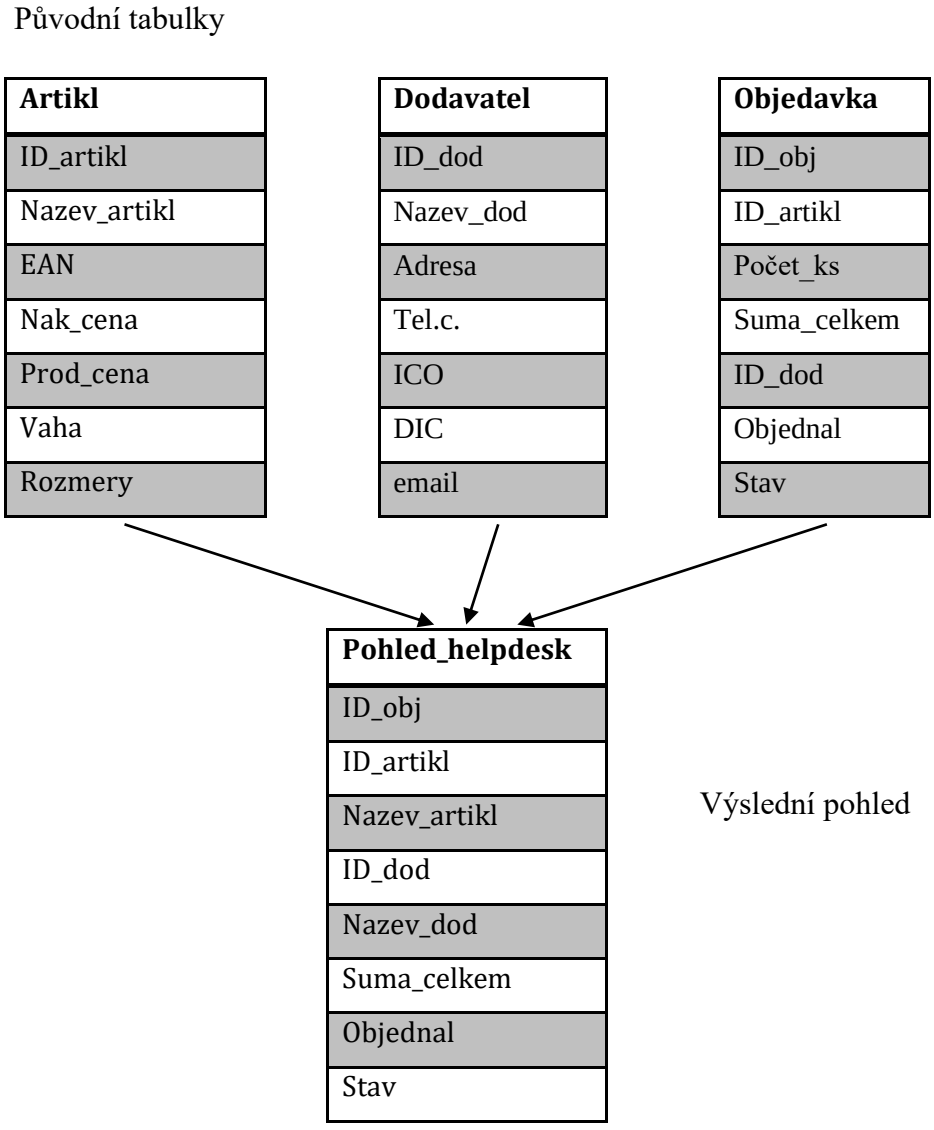
Vytváření pohledů

Abychom mohli vytvářet pohledy ve svém schématu, potřebujeme oprávnění create view. Oprávnění create any view nám umožní vytvářet pohledy i ve schématech jiných uživatelů. Abychom mohli vytvořit pohled, potřebujeme mít pro zamýšlené činnosti, tj. select, update, insert nebo delete přímá oprávnění k tabulkám nad kterými budeme pohled vytvářet. Nelze úspěšně vytvořit pohled, pokud má uživatel k objektům oprávnění přes role. [10]

Tabulka 4 Pohledy. Zdroj [10]

Syntaxe	Popis
CREATE OR REPLACE	Je-li pohled vytvářen s dodatkem or replace a pokud už existuje pohled stejného jména, bude původní pohled nahrazen novou definicí. Původní oprávnění zůstanou beze změny
VIEW	Klíčové slovo, které říká Oracle, že má vytvořit pohled
SCHEMA.VIEW	Jméno pohled. Pokud neuvedeme schíma,

	je poheld vytořen v aktuálním schématu
AS <DOTAZ>	Definice dotazu, na kterém je pohled založen. V poddotazech může být použito až 1000 příkazů
KLAUZULE_WITH	Použití pohledu může být omezeno následovně: READ_ONLY – touto kluzulí zabránime, aby před pohled bylo možné vkládat záznamy, modifikovat data nebo odstranit záznamy z tabulek, nad kterými je pohled vytvořen



Obrázek 4 Využití pohledu pro omezený přístup k sloupcům – vlastní tvorba

Syntaxe je následující:
SQL> CREATE OR REPLACE VIEW jméno_pohledu AS dotaz;

Oprávnění pro pohledy se podobají oprávněním pro tabulky. Řádky v pohledu je možné vybírat, aktualizovat, odstraňovat nebo vkládat, samozřejmě za předpokladu, že pohled je aktualizovatelný. Aby mohli pohled používat i ostatní uživatelé, musí mít vlastník pohledu přidělena oprávnění pro základní tabulky s klauzulí GRANT OPTION nebo musí mít systémové oprávnění ADMIN OPTION. [12]

4. Vlastní práce

4.1. Zmapování momentální situace řešené problematiky

Zmapovat momentální situaci ve firmě je moc těžké. Firma vás ke svým datům jen tak nepustí, jak s daty jako zaměstnanec nepracujete je velice malá pravděpodobnost, že by vám byla databázová aplikace nainstalována, nebo jste dostali jen tak přístupová práva. Proto je toto téma velice obtížné zmapovat.

Momentální situace ve firmách je různá. Někde jsou databáze zabezpečeny heslem i přístupovými právy, někde jenom přístupovými právy a někde si uživatel databázi na svém PC otevře a pracuje s dotazy bez předešlého přihlášení a přístupových práv. Právě tady vzniká špatná dohledatelnost úprav dat v databázi. [13]

Na začátku je nutný identifikovat a analyzovat informace a data, které se musí ochránit. K tomuto kroku je potřebné znát logiku a architekturu databáze. Je to hlavně z toho důvodu, aby se správně identifikovali citlivá data a jejich umístění. Ne všechna data jsou riziková, a ne všechna data třeba chránit. [13]

Prvním krokem před využitím ochranných technik a nástrojů je analyzovat a identifikovat důležité informace, které musí být chráněny. K tomu je zapotřebí rozumět logice a architektuře databáze, díky tomu pak budete moci snadněji rozhodnout, kde a jaká citlivá data chránit. [13]

Dále je vhodné vést záznamy o všech databázích ve firmě, předejdeme tak možným ztrátám informací a také nám to usnadní zálohování. Když citlivá a důvěrná data vyčleníme, využijme robustní algoritmy šifrování. Jestliže, by útočníci získali přístup k serveru nebo systému, první věc, kterou by zkusili ukrást, by pravděpodobně byla databáze. Nejlepším způsobem, jak databázi zabezpečit, je zamezit tomu, aby v ní nepovolaní uživatelé mohli číst. [13]

Mnoho společností investuje čas a zdroje na ochranu vlastních produktivních databází, ale při vývoji projektu nebo vytváření testovacího prostředí jednoduše duplikují původní databázi a používají ji v prostředí, které je mnohem méně kontrolované. Citlivé údaje jsou v takovém případě zbytečně vystavované riziku odcizení. [13]

Maskování nebo anonymizování je proces, při kterém je vytvořena podobná verze se stejnou strukturou jako originál, ale citlivá data jsou upravena tak, aby byla v bezpečí. Mění se hodnota, ale formát zůstává stejný. Data je možné měnit různými způsoby: zamícháním, šifrováním nebo zamícháním znaků či slov. Zvolená metoda závisí na administrátorovi. Zaznamenáváním akcí a pohybů dat budeme vědět, se kterými informacemi se hýbalo, ale také kdy a kým. Kompletní historie transakcí nám umožní porozumět přístupům k datům či jejich úpravám. Při správě databází na tyto tipy pamatujme a mějme na vědomí, že pro útočníky jsou databáze velice lákavým terčem a jejich zabezpečení si naši pozornost zaslouží. [13]

4.2. Bezpečnostní rizika

Než se budeme podrobněji věnovat možným rizikům, je potřebné si uvědomit, jaká rizika společnosti hrozí při nezabezpečené databázi. Jaké data shromažďuje a jaká rizika mohou společnosti vzniknout, pokud nebudou bezpečností rizika nastaveny správně. Samozřejmostí je také proškolení uživatelů.

- Krádež a zpronevěra
- Ztráta utajení
- Ztráta soukromí
- Ztráta integrity
- Ztráta dostupnosti [14]

Nejčastější problémem se samotným provozem databáze je nevědomost. Většina společností, databáze vůbec nesleduje. Dodavatel nainstaluje vlastní aplikaci i s databází a tím to končí. Nikdo pak neví, co se v databázi děje a až nastane problém, pak se všichni diví a narychlo se hledá někdo, kdo to opraví. Databáze se dá lehko poškodit a je dokázáno, že nejčastěji data z databáze odcizí nebo poškodí interní zaměstnanec. [16]

Takže nejsou-li správně nastavena oprávnění přístupu na databázový server, potažmo do databáze, pak může téměř kdokoliv přistupovat k citlivým údajům. Je potřeba zkontrolovat jednotlivé účty, hesla, práva. [16]

Důležité je také auditování, tedy sledování, kdo a kam přistupoval. Kdo pracuje s citlivými údaji, měl by to mít podle standardů. Data se pak mohou i různě šifrovat, ale to je na delší povídání. [14]

Poslední dobou se stále častěji odborníci setkávají s uživateli, kteří přehlíží anebo vůbec nevědí, že databáze má svůj vlastní systém uživatelských práv. Do značné míry je to zapříčiněno tím, že webhosting vám již přidělí účet a práva na používání vaší databáze, kde již nemůžete žádné účty vytvářet. Můžete se však lehce dostat do situace, kdy po vás vedení bude chtít, abyste ve firmě realizovali databázový server, který bude určen pro více aplikací. A v tomto případě nemůžete ukládat data z více aplikací do jednoho účtu kvůli přehlednosti. Pro tyto a další situace má databázový systém možnost využití uživatelských práv. [14]

Nejdříve si musíme nadefinovat pojem **privilegium** – je to oprávnění provádět určitou činnost s určitým objektem a je vázána na konkrétního uživatele. Vždy při vytváření uživatele v databázi mu musíte přidělit privilegia (práva). S vytvářením nových uživatelů se váže pojem **princip nejnižšího oprávnění**. Tento princip se aplikuje především kvůli zvýšení bezpečnosti databáze. Princip nejnižšího oprávnění říká, že: *uživatel nebo procedura by měl mít nejnižší možná oprávnění, která mu umožní provést patřičný úkol*. Do praxe převedeno: nebudete přidělovat uživatelům, kteří potřebují data z databáze pouze zobrazit pravomoc DROP, která může databázi nebo tabulku smazat. [14]

Pro přidělování práv máme v databázi příkaz GRANT a REVOKE pro odebírání. Tyto příkazy fungují na čtyřech úrovních: [14]

- Global – globální
- Database – pro určitou databázi
- Table – pro určitou tabulku

- Column – pro určitý sloupec

Každému novému uživateli můžeme přidělit poměrně mnoho práv, ne všechna jsou však vhodná pro běžné uživatele. Abychom zachovali princip nejnižšího oprávnění, je vhodné vytvořit uživatele bez práv a po konzultaci s ním mu potřebná práva nastavit. Toto řešení však nelze aplikovat vždy, proto můžeme rozdělit práva na: [14]

- uživatelská
- administrátorská
- zvláštní [14]

Tabulka 5 Práva uživatelů. Zdroj [14]

Právo	Vztahuje se na	Popis
SELECT	tabulky, sloupce	Povoluje vybírat záznamy z tabulek.
INSERT	tabulky, sloupce	Povoluje vkládat nové řádky do tabulky.
UPDATE	tabulky, sloupce	Povoluje obnovovat hodnoty záznamů v tabulkách.
DELETE	tabulky	Povoluje odstraňovat záznamy (řádky) z tabulek.
INDEX	tabulky	Povoluje vytvářet a odstraňovat indexy z tabulek.
ALTER	tabulky	Povoluje měnit strukturu stávajících tabulek (přejmenovávat nebo přidávat sloupce, měnit datové typy sloupců).
CREATE	databáze, tabulky	Povoluje vytvářet nové databáze a tabulky.
DROP	databáze, tabulky	Povoluje odstranit databáze a tabulky.

Při návrhu databáze musíme mít na paměti, pro kolik uživatelů databázi navrhujeme a jak bude databáze velká. Pokud máme málo uživatelů, málo dat (do několika málo miliónů řádků), tak nám realita hodně chyb promine. Pokud máme mnoho dat, mnoho uživatelů, tak pak chyby v návrhu většinou způsobují obtížné provozní problémy a většinou i problémy při údržbě a dalším rozšiřování software. [15]

5. Návrh konkrétního řešení

V modelové společnosti zaměstnanci pracují v zastaralém systému, do kterého se zaměstnanci přihlašují svým heslem, nebo si heslo půjčují. V nejhorším případě se nepřihlašují vůbec, protože je to pro ně zbytečný a pracují mimo firemní systém například v excelové tabulce. To je pro správní práci s daty a jejich využití v celé společnosti nepřijatelné. Data se nedají dohledat, jejich aktuálnost a správnost je nejistá. Dále se nedá dohledat případná změna dat, nebo jejich úplná ztráta.

Dané společnosti bude navrhnut nový systém, který si zobrazíme jen okrajově a naplníme jej daty. Nejdůležitější částí bude popis jednotlivých pracovních pozic, které do systému budou mít přístup. Dále budeme muset podrobně přejít přístupové role, práva a možné pohledy.

V novém návrhu je velice důležité, aby daná společnost zapojila do tvorby své zaměstnance, kteří s daty jakkoli pracují. Uživatelé se musí motivovat k práci v novém systému, tato část práce je zezačátku také na společnosti, která nový systém zavádí, ale tak na zaměstnavateli. Motivace musí být přímá a musí být vidět, že novým systémem si uživatel polepší. Musí být jasně ukázáno, že nový systém bude jednoduchý na obsluhu. Ze strany zaměstnavatele se musí uživatelé ubezpečit, že se začne pracovat s novým systém až po jeho kompletním zavedení do společnosti, kdy nový systém bude plně funkční bez zásahu dodavatelské společnosti. Dále pak po kompletním proškolení všech uživatelů. Proto je velice důležité, aby se zaměstnanci podíleli na popisu jednotlivých přístupů, protože oni jsou zadavateli dat do systému a zaměstnanci s těmito daty dále pracují a využívají je ke své práci.

Po samotném nastavení zabezpečení databáze na základě popisu od zaměstnanců a splnění dohody s vedením se dostáváme do poslední fáze tzv. projektu a to testování. Testování v reálném projektu provádí zaměstnanci neboli konkrétní uživatelé. V dané kapitole se provedou jednoduché testy, které odhalí chyby nebo správní zabezpečení přístupu do databáze.

5.1. Popis modelové společnosti

Modelová společnost působí na českém trhu již od roku 2004 a zabývá se maloobchodem potravinového a nepotravinového zboží. Vlastní nebo pronajímá prostory, v kterých má své obchody. V současnosti má otevřené tři obchody v Praze a další 4 po celé republice.

Společnosti se na trhu daří i v této nelehké době, a to díky tomu, že její předností jsou potraviny, které zpracovává sama, jako je pečivo nebo maso a masové výrobky.
Rozmanitost zboží je veliká, a proto svým zákazníkům nabízí téměř vše.
Od svého vzniku společnost stabilně roste, jednotlivé obraty jsou dostatečné, a proto je společnost v zisku. Konkurenční společnosti se snaží předhánět se v nabízených službách nebo v rozmanitosti jednotlivého zboží.

Vize společnosti pro následující obchodní období spočívá v zavedení funkční databáze do provozu. Zavedené nové databáze do systému celé společnosti je nákladní ale data jsou pro danou společnost velice cenou komoditou. Proto je změna nutná, aby zaměstnanci přesně věděli, jaká data zpracovávají, k čemu slouží a co by znamenalo daná data ztratit. Zabezpečení a následné přesné zobrazení například změny dat v databáze je nutností. K tomu musí být jednotlivý zaměstnanci také proškoleny, nestačí jenom dát každému uživateli roli nebo práva, také musí vědět, co se zatím skrývá a co všechno se tím dá v databázi ovlivnit a také následně odhalit. Jak již bylo zmíněno, uživatelé nového systému se musí řádně proškolit a musí si být vědomí toho, jaká rizika vznikají společnosti při jakékoliv nekalé činnosti s daty.

5.1.1. Organizační struktura modelové společnosti

Struktura společnosti je rozdělena do několika úseků neboli větví, které jsou popsány a zobrazeny na obrázku.

Obrázek 5 Organizační struktura společnosti – vlastní tvorba

Vysvětlení zkratky PP – procesní podpora

Vedení společnosti nemývá přístup do databáze, protože s ní nepracuje, ale vyžaduje z ní výstup. Daným výstupem se rozumí například report. Tvorba reportů není součástí diplomové práce, a proto se pro vedení společnosti vytvoří pohled. Daný pohled musí být ve své podstatě přehled obratů jednotlivých nákupčích. Je to přehled jejich práce.

Personální oddělení zpracovává data o zaměstnancích. Uživatelé pracují s oprávněním zakládat nové údaje při nástupu nového zaměstnance do společnosti, dále mají oprávnění data o zaměstnancích měnit. V daném případě by šlo o změnu osobních údajů, data o mzdě nebo zařazení v rámci společnosti. Také evidují termín nástupu a oznamují výročí daného zaměstnance. Uživatelé personálního oddělení nemají oprávnění daného zaměstnance z evidence vymazat v případě ukončení pracovního poměru. V tomto případě je nutný zásah správce databáze.

Oddělení Nákupu je oddělení, které spravuje data o artiklech. Toto oddělení na začátku domlouvá obchodní podmínky s dodavatelem. Tyto obchodní podmínky obsahují data o artiklech, nákupních cenách, dále prodejnách pro dodací podmínky a v neposlední řade smluvní pokuty za nedodávky. Po úspěšném uzavření smlouvy oddělení Nákupu posílá smlouvu na Fakturační oddělení, kde budou data o dodavateli založena do systému. Po vzniku ID dodavatele může oddělení Nákupu založit nový artikl do systému.

Založí mu název, údaje o cenách, kde nákupní cena musí být vždy uvedená a prodejní cena se může zadat dodatečně. Je to kvůli konkurenci, a aby nebylo založeno zbytečné množství nesprávních cen i kvůli ČOI. Všechny tyto potřebná data obdrží nákupčí od dodavatele. Tyto údaje se musí potvrdit z obou stran, hlavně údaje typu nákupní cena, dále také, na které prodejně se bude zboží prodávat. Tento údaj je nutný k dodržení obchodních podmínek a zároveň je stanovena smluvní pokuta za nedodávku na předem nasmlouvané prodejny. Je nutné, aby byly dodrženy obchodní podmínky a výše nákupních cen. Následně se všechny data založí do databáze a vygeneruje ID_artiklu. Toto oddělení zpracovává v databázi také objednávky. Objednávky jsou ve vztahu prodejna tedy odběratel a dodavatel. Výše objednávky je buďto dána smlouvou nebo odprodeji.

Fakturační oddělení má oprávnění zpracovávat data o dodavatelích, odběratelích a fakturách. Data ve jmenovaných tabulkách můžou uživatelé číst, měnit, zakládat a mazat. Mazat má ale oprávnění jenom jedna osoba v oddělení. Nebo se můžou z fakturace s dotazem obrátit na správce nebo na procesní podporu. Data o odběratelích fakturační oddělení zakládá jenom v případě, kdy se otevře nová prodejna. Tyto údaje jsou neměnné až na telefonní číslo. V případě uzavření obchodu může fakturační oddělení tuto informaci evidovat, ale data ze systému se neupravují.

Oddělení Marketingu zpracovává reklamní kampaně, aby byla společnost stále v povědomí všech potencionálních zákazníků. Data z jednotlivých tabulek nemají k dispozici ani ke čtení. Pro toto konkrétní oddělení se v databázi vytváří pohledy. Tyto pohledy jsou pro marketing vodítkem ke zpracování potřebných údajů a přípravě letákové akce, nebo TV reklamy. Můžou se zaměřit na jednotlivé zboží a propagovat kampaň na vybrané artikly, nebo se také můžou zaměřit na propagaci samotné prodejny, a to z důvodu např. oslavy výročí prodejny.

Oddělení Procesní podpory je důležitým oddělením a jeho hlavním úkolem je v první řadě podpora. Podporují oddělení jako nákup, účetní, marketing a IT. Pro oddělení IT je oddělení procesní podpory jako mezi článek pro komunikaci s ostatními odděleními. Při zavádění nové databáze, se projektu účastní odd. IT a také odd. procesní podpory. Po zavedení nové databáze do provozu již zpracovává běžné dotazy na funkcionalitu právě

oddělení procesní podpory. Při zavedení nové databáze do provozu nebo nového zabezpečení zaměstnanci tohoto oddělení spolupracují s ostatními odděleními a snaží se zamezit chybovosti a problémům které vznikají při změně zpracování dat v databázi. Uživatelé procesní podpory mají vysoké oprávnění v databázi, a proto v mnohých případech upravují nebo mažou data z databáze. Toto oddělení se také podílí na zpracování potřebných podkladů ke školení. Samotné školení probíhá ve spolupráci s oddělením IT. Kde procesní oddělení zastřešuje komunikaci mezi odděleními a pomáhá uživatelům při denní činnosti.

Oddělení IT zpracovává požadavky od uživatelů jenom v takových případech, kde uživatelé nemají dostatečné oprávnění v databázi na dané úkony. Toto oddělení zpracovává také technické požadavky, a to od celé společnosti. Má na starosti jak nákup výpočetní techniky, tak správu této techniky a její bezproblémový chod. V případě nefunkčnosti samotného systému se jednotlivá oddělení obrací na IT. Mají na starosti také zálohování systému, takže v případě poškození dat uživateli může tuto chybu odstranit zálohou. Nespravují ale samotná data, to má na starosti oddělení Procesní podpory. IT oddělení ale zpracovává požadavky z oddělení procesní podpory. Jsou to požadavky na změny v tabulkách, kde oddělení PP nemá přístup.

V modelové společnosti bude založená nová databáze a následně bude daná databáze zabezpečená. Na začátku se zpracují jednotlivé tabulky podle požadavků od modelové společnosti, tak aby byla dodržena relace mezi jednotlivými tabulkami. Dále se dodrží normalizace dat. V neposlední řadě se databáze zabezpečí, a to tvorbou uživatelských účtu, profilem, rolemi a pohledy.

5.1.2. Přehled tabulek

Níže jsou popsány postupy při tvorbě jednotlivých tabulek modelové společnosti. Pro zobrazení je vytvořeno jen několik tabulek, pro které se bude nastavovat zabezpečení. Vytvořené tabulky budou naplněny daty, a nakonec jednoduchým selectem budou zobrazeny data v jednotlivých tabulkách.

- **Tabulka artikl**

```
CREATE TABLE artikl (
    ID_artikl CONSTRAINT artikl_pk primary key,
    nazev varchar(150) not null,
    cena_NC numeric not null,
    cena_PC numeric,
    ID_dod integer,
    nazev_dod varchar(100),
    ID_zam integer not null,
    nazev_odd varcharg(100) not null,
    ID_odb integer not null,
    FOREIGN KEY (ID_dod) REFERENCES dodavatel (ID_dod)
    FOREIGN KEY (ID_zam) REFERENCES zamestnanec (ID_osoba)
    FOREIGN KEY (ID_odb) REFERENCES odberatel (ID_odb)
 );

INSERT INTO artikl VALUES (1, avokado, 31.21, 49.9, 25896,
Kupovozel, 7894, cerstve, 001);

1 row inserted.
INSERT INTO artikl VALUES (2, mango, 28.23, 49.9, 25896,
Kupovozel, 7894, cerstve, 001);

1 row inserted.
```

```
SELECT * from artikl;
```

Tabulka 6 Tabulka artikl – vlastní tvorba

ID_artikl	nazev	cena_NC	cena_PC	ID_dod	nazev_dod	ID_zam	nazev_odd	ID_odb
1	avokado	31.21	49.9	25896	Kupovozel	7894	cerstve	001
2	mango	28.23	49.9	25896	Kupovozel	7894	cesrstve	001
3	magnesia	6,12	15.9	24799	Voda	7412	napoje	002
4	milka	28.12	28.9	23691	Coko	7532	sladke	001
5	nivea	19.31	25.9	22477	Krasa	7643	drogerie	002

- **Tabulka zaměstnanec**

```
CREATE TABLE zamestnanec (
    ID_osoba constraint zamestnance_pk primary key,
    jmeno varchar (50) not null,
    prijmeni varchar (100) not null,
    adresa varchar (150) not null,
    mesto varchar (100),
    tel_c integer,
    ID_pozice integer not null,
    ID_odd integer not null,
    plat integer,
    FOREIGN KEY (ID_pozice) REFERENCES pozice (ID_pozice)
    FOREIGN KEY (ID_odd) REFERENCES oddeleni (ID_odd)
);

INSERT INTO zamestnanec VALUES (7894, Adela, Mala, Chrtova
15, Praha, 771235689, 1, 1, 29000);

1 row inserted.
```

```
INSERT INTO zamestnanec VALUES (7412, Petr, Drobny, Certova
1, Praha, 772456789, 2, 2, 45000);

1 row inserted.

SELECT * from zamestnanec;
```

Tabulka 7 Tabulka zaměstnanec – vlastní tvorba

ID_osoba	jmeno	prijmeni	adresa	město	tel_c	ID_poz	ID_odd	plat
7894	Adela	Mala	Chrtova 15	Praha	771235689	1	1	29000
7412	Petr	Drobny	Certova 1	Praha	772456789	2	2	45000
7532	Marek	Ptak	Parizska 2	Praha	773485685	2	3	40000
7643	Milada	Novakova	Konecna 1	Praha	774786453	1	5	39000

- **Tabulka pozice**

```
CREATE TABLE pozice (
    ID_pozice constraint pozice_pk primary key,
    nazev_pozice varchar(150) not null
);

INSERT INTO pozice VALUES (1, nakupci);

1 row inserted.

SELECT * from pozice;
```

Tabulka 8 Tabulka pozice – vlastní tvorba

ID_poz	nazev_pozice
1	nakupci
2	asistent_n
3	ucetni
4	proces_podpora

- **Tabulka oddělení**

```
CREATE TABLE oddeleni (
    ID_odd constraint oddeleni_pk primary key,
    nazev_odd varchar (100) not null
);

INSERT INTO oddeleni VALUES (1, cerstve);

1 row inserted.

SELECT * from oddeleni;
```

Tabulka 9 Tabulka oddělení – vlastní tvorba

ID_odd	nazev_odd
1	cerstve
2	napoje
3	sladke
4	slane
5	drogerie

- **Tabulka dodavatel**

```
CREATE TABLE dodavatel (
    ID_dod constraint dodavatel_pk primary key,
    nazev_dod varchar (100) not null,
    ulice varchar (100) not null,
    mesto varchar (50),
    stat varchar (50),
    DIC integer,
    ICO integer,
    tel_c integer
);
```

```
INSERT INTO dodavatel VALUES (25896, Kupovozel, Ovicka 25,
Praha, CZ, CZ1234567890, 12345678, 605111112);

1 row inserted.

INSERT INTO dodavatel VALUES (24799, Voda, Okruzna 2, Plzen,
CZ, CZ8754212366, 87451236, 772145125);

1 row inserted.

SELECT * from dodavatel;
```

Tabulka 10 Tabulka dodavatel – vlastní tvorba

ID_dod	nazev_dod	ulice	město	stat	DIC	ICO	tel_c
25896	Kupovozel	Ovicka 25	Praha	CZ	CZ1234567890	12345678	605111112
24799	Voda	Okruzna 2	Plzen	CZ	CZ8754212366	87451236	772145125
23691	Coko	Prazska 18	Ostrava	CZ	CZ1111111113	85236541	778132144
22477	Krasa	Kratka 1	Brno	CZ	CZ8522588522	65852135	777125521

- **Tabulka odberatel**

```
CREATE TABLE odberatel (
    ID_odb constraint odberatel_pk primary key,
    nazev_odb varchar (30) not null,
    adresa varchar (50) not null,
    mesto varchar (30),
    tel_c integer
);

INSERT INTO odberatel VALUES (001, Cakovice, Okruzna 25,
Praha, 608123951);

1 row inserted.
```

```
INSERT INTO odberatel VALUES (002, Zlicin, Prazska 12, Praha,
608782149);

1 row inserted.

SELECT * from odberatel;
```

Tabulka 11 Tabulka odběratel – vlastní tvorba

ID_odb	nazev_odb	adresa	mesto	tel_c
001	Cakovice	Okruzna 25	Praha	608123951
002	Zlicin	Prazska 12	Praha	608782149

- **Tabulka objednávka**

```
CREATE TABLE objednavka (
    ID_obj constraint objednavka_pk primary key,
    ID_artikl integer not null,
    nazev_artikl varchar (100) not null,
    ID_dod integer,
    nazev_dod varchar (100) not null,
    cena_NC integer,
    datum date
    ID_odb integer,
    FOREIGN KEY (ID_artikl) REFERENCES artikl (ID_artikl)
    FOREIGN KEY (ID_dod) REFERENCES dodavatel (ID_dod)
    FOREIGN KEY (ID_odb) REFERENCES odberatel (ID_odb)
);

INSERT INTO objednavka VALUES (12356, 1, avokado, 25896,
Kupovozel, 31.21, Date, 001);

1 row inserted.
```

```
SELECT * from objednavka;
```

Tabulka 12 Tabulka objednávka – vlastní tvorba

ID_obj	ID_artikl	nazev_artikl	ID_dod	nazev_dod	cena_NC	datum	ID_odb
12356	1	avokado	25896	Kupovozel	31.21	2020-09-11	001
13345	3	magnesia	24799	Voda	6.12	2020-10-12	001
13542	5	nivea	22477	Krasa	19.31	2020-09-01	002

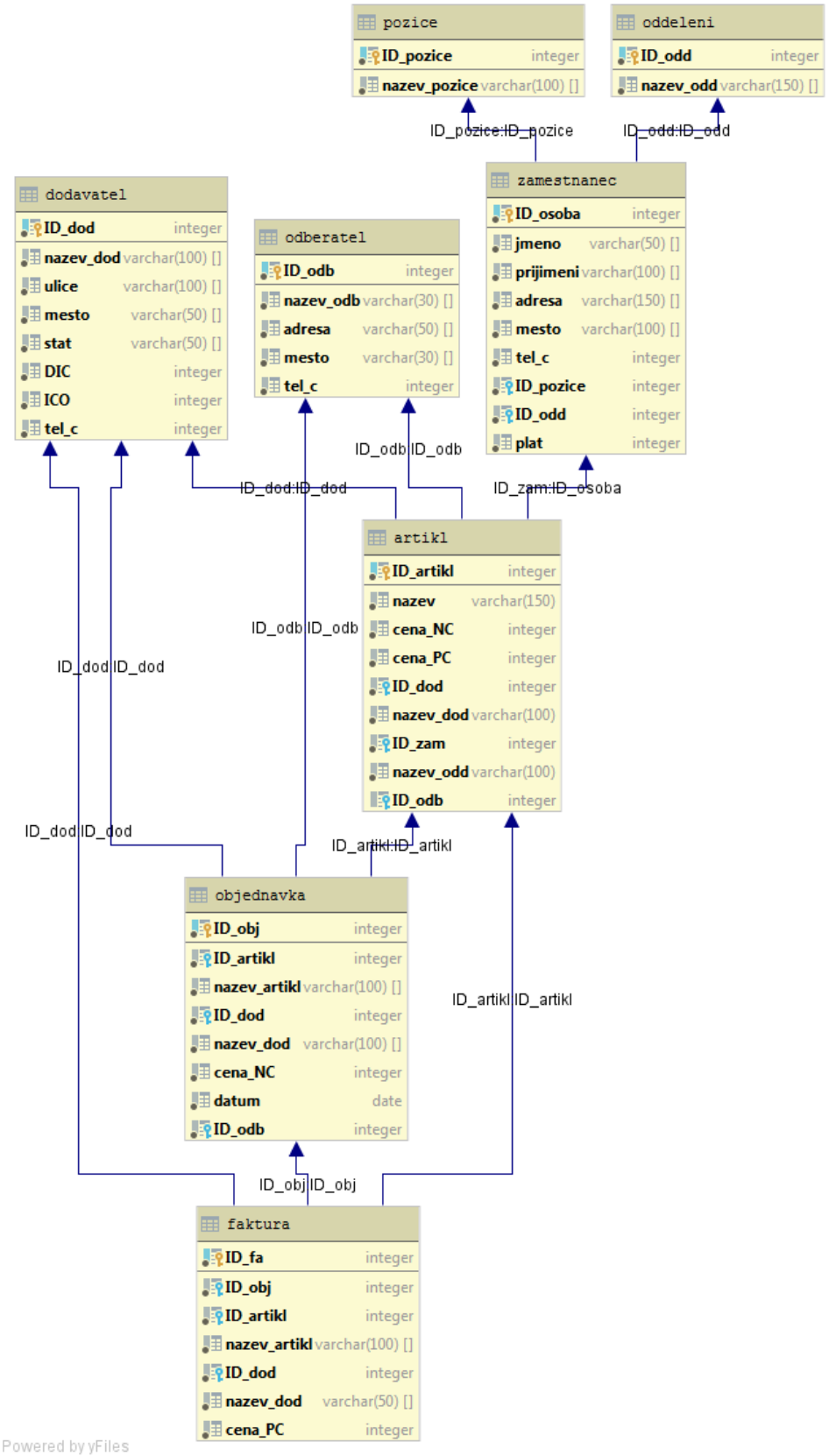
- **Tabulka faktura**

```
CREATE TABLE faktura (
    ID_fa constraint faktura_pk primary key,
    ID_obj integer not null,
    ID_artikl integer not null,
    nazev_artikl varchar(100)  not null,
    ID_dod integer,
    nazev_dod varchar(50) not null,
    cena_PC integer,
    FOREIGN KEY (ID_obj) REFERENCES objednavka (ID_obj)
    FOREIGN KEY (ID_artikl) REFERENCES artikl (ID_artikl)
    FOREIGN KEY (ID_dod) REFERENCES dodavatel (ID_dod)
);

INSERT INTO faktura VALUES (45871, 12356, 1, avokado, 25896,
Kupovozel, 49.9);

1 row inserted.
SELECT * from faktura;
```

Tabulka 13 Tabulka faktura – vlastní tvorba

ID_fa	ID_obj	ID_artikl	nazev_artikl	ID_dod	nazev_dod	cena_PC
45871	12356	1	avokado	25896	Kupovozel	49.9
48741	13345	3	magnesia	24799	Voda	15.9
46871	13542	5	nivea	22477	Krasa	25.9

5.1.3. Model – ERD



Obrázek 6 Model ERD – vlastní tvorba

5.2. Zabezpečení databáze

V modelu jsou zobrazeny jenom některé tabulky, které postačí pro zobrazení možného zabezpečení databáze v společnosti. K jednotlivým tabulkám budou nastaveny přístupy podle popisu z jednotlivých oddělení. V reálním světě se tvorby zabezpečení neboli přístupu do databáze zúčastňují samotní uživatelé. Pomocí dotazování na schůzkách se vývojář databáze dozví, co jednotlivý uživatelé potřebují mít přístupné. Kam se v databázi konkrétní uživatelé musí dostat a kam zase vůbec ne. Co všechno potřebují k výkonu své práce mít povolené.

V projektu se musí dodržovat určitá posloupnost kroků a k jejich naplnění je nutný určitý čas. V tomto kroku se uskutečnili pohovory s uživateli z jednotlivých oddělení. Pohovorů se účastnili samotní uživatelé databáze, ti, který pracovali ve staré databázi, jejich nadřízení, zaměstnanci procesní podpory a oddělení IT. Střetnutí by probíhalo formou rozhovorů a dotazování na práci s databází, dále dotazování na práci s daty a jejich další využití.

Výstupem musí být po určitém času jednoznační přehled všech uživatelů, jejich práva a povinnosti. Právy je myšleno, přístup do databáze, změna struktury databáze nebo změna samotných dat. Povinnosti uživatele vyplývají z jeho pracovního zařazení. Toto všechno musí být zaznamenáno a daný výstup musí být jednotlivými odděleními odsouhlasen. V případě, že se později najdou nesrovnalosti mezi tím, co je nastaveno v databázi a požadavky od uživatelů, stačí tuto nesrovnalost zkontrolovat ve výstupním dokumentu. Jakmile došlo k pochybení na straně vývojáře, chyba se okamžitě odstraní. Může se ale stát, že uživatelé na něco zapomněli a v tom případě se to řeší podle smlouvy mezi společností a dodavatelskou společnosti, která má zabezpečit databázi, ale této problematice se nebudeme věnovat.

5.2.1. Uživatelský účet

Prvním krokem k zabezpečení přístupu do databáze je nutné vytvořit uživatelské účty. Jakmile se bude moct uživatel přihlásit svým heslem do databáze, je namístě mu přidělit oprávnění na práci s databází a samotnými daty.

Vytvoření nového uživatelského účtu a zároveň nutnost uživatele změnit si heslo při prvním přihlášení.

```
GRANT USER novak IDENTIFIED BY heslo PASWORD EXPIRE;
User created.
```

Tímto způsobem by se vytvářeli přístupy v databázi. V tomto konkrétním případě bude muset pan Novák zadat heslo při každém přihlášení do databáze. V rámci zabezpečení databáze je toto první krok, který se musí dotáhnout pro každého uživatele. Je tu ještě jedna možnost a to taká, že se uživatel nebude muset přihlašovat heslem do databáze a databázový systém se spolehne na přihlašovací jméno a heslo použito při přihlášení do systému. Pro ilustraci by uživatelský účet, který by měl přístup do databáze, a mohl upravit data, vypadal takto.

```
GRANT select, update ON artikl TO novak;
```
Přidělení práv na select a update panu Novákovi a na tabulku artikl.

```
GRANT ALL PRIVILE ON artikl TO novak;
```
Přidělení všech práv na tabulku artikl panu Novákovi.

```
GRANT ALL ON artikl TO novak WITH GRANT OPTION;
```
Přidělení všech práv s možností přidělovat práva na tabulku artikl.

Příklady zobrazují možnosti práce s uživatelským účtem. Modelová společnost je velká, co se týče počtu zaměstnanců. Proto není vhodné nastavovat práva pomocí uživatelského účtu.

5.2.2. Nastavení profilu

Dalším krokem bude nastavení Profilu na uživatelské účty. Nově vzniknutým uživatelským účtům se musí přiřadit profil. Profil se vytvoří jeden a ten se přiřadí všem uživatelským účtům.

```
CREAT PROFILE persona LIMIT
     FAILED_LOGIN_ATTEMPTS         3
     PASSWORD_LOCK_TIME            .1
     PASWWORD_LIFE_TIME            90
     PASSWORD_GRACE_TIME           15
     PASSWORD_REUSE_TIME           60
     PASSWORD_REUSE_MAX            UNLIMITED
     PASSWORD_VERIFY_FUNCTION      VERIFY_FUNCTION;
```

Tímto příkazem se vytvořil profil PERSONA, který definuje, že po třech neúspěšných pokusech se přihlásit se účet zamkne a po jedné hodině se ale účet opět odemkne. Doba platnosti hesla je stanovena na 90 dní a dalších 15 dní je možné toto heslo změnit. Stejné heslo je možné použít až po uplynutí 60 dní. Druhý parametr, který sleduje taktéž historii hesel je nastaven na unlimited. Dané parametry se vylučují, což znamená, jestliže má nastaven jeden parametr určitou hodnotu tak druhý musí mít unlimited. V tomto případě je lepší nastavit unlimited na password_reuse_max protože to by mohl zkušenější uživatel obejít a to tak, že by měnil tolikrát heslo až by se mu podařilo znova si uložit původní heslo.

Uživatelský učet založen s profilem na kontrolu hesla.

```
CREAT USER Novak IDENTIFIED BY heslo PASWORD EXPIRE PROFILE
persona;
```

5.2.3. Role

Daná kapitola se bude věnovat nastavením rolí. Role je ve své podstatě uživatelský účet s oprávněním pro větší počet lidí. Je to pro správce databáze přijatelnější způsob práce a také jednoduší a přehlednější při přidání nebo odebrání práv.

Prvním krokem bude na kombinovat uživatelé tak, aby nově založené role vyhovovali uživatelským potřebám. Jejich počet se na začátku nedá odhadnout, ale musí splnit požadavky a zároveň by jich nemnělo být příliš vela kvůli přehlednosti.

Přehled přístupových práv v rámci modelové společnosti po jednotlivých odděleních a uživatelích by byl veliký, a proto pro ilustraci jsou zobrazeny jenom některé pozice v tabulce 12.

Tabulka 14 Přehled práv – vlastní tvorba

	User		Tabulky						
		Zaměstnanec	Oddělení	Pozice	Artikl	Dodavatel	Odběratel	Objednávka	Faktura
nákup	**Novák**	-	s	s	all	s,u	-	all	s
	Vyskočilová	-	s	s	all	s,u	-	all	s
	Kolaříková	-	s	s	all	s,u	-	all	s
účet	**Zátopek**	s	s	s	-	all	all	s,u	all
	Pokorný	s	s	s	-	s,u	s,u	s,u	s,u
	Malá	s	s	s	-	s,u	s,u	s,u	s,u
IT	**Pták**	all	all	all	all	all	all	all	all
	Kyselý	all	all	all	all	all	all	all	all
PP	**Drobný**	-	all	all	all	all	all	all	all
	Nováková	-	all	all	all	all	all	all	all
pers	**Poláková**	all	all	all	-	-	-	-	-
	Koberec	s,i,u	s,i,u	s,i,u	-	-	-	-	-

Vysvětlení zkratek:
Select – s
Insert – i
Update – u
Delete – d

V tomto momentě již máme vytvořené uživatelské účty pro uživatelé, kteří pracují v databázi. Z tabulky nám vyplývá, že se musí nastavit minimálně čtyři role, a to pro oddělení nákupu, účetní oddělení, personální oddělení a oddělení procesní podpora. Uživatelé z oddělení IT jsou správci databáze, jim bude také vytvořena role.

- **Nákupní oddělení**

Z výše uvedené vzorové tabulky si můžeme vzít informaci a vytvořit roli. Role se bude jmenovat „nakup“. Půjde o roli, která se nastaví všem zaměstnancům v oddělení nákupu. Rozhodovalo se, jestli není namístě rozdělit oprávnění ještě na vedoucí a asistent. V tomto oddělení je to nastaveno tak, že asistenti zpracovávají v databázi vše, co se týče artiklů v tabulce artikl, takže vzít jim oprávnění na delete a ponechání tohoto práva jenom na vedoucí by znemožnilo běžný chod práce. Oddělení nákupu může zpracovávat také data v tabulkách odděleni, pozice a faktura. K daným tabulkám má oddělení právo jenom číst. Tyto data potřebují k správnímu zpracování dat v tabulce artikl a objednávka. Při zakládání artiklu do databáze musejí být vyplněny údaje o nákupčím.

```
CREATE ROLE nakup;
GRANT nakup TO novak, vyskocilova, kolarikova;
GRANT select ON oddeleni, pozice, faktura TO nakup;
GRANT all ON artikl, objednavky TO nakup;
GRANT select, update ON dodavatel TO nakup;
```

- **Fakturační oddělení**

Fakturační oddělení zpracovává data z tabulek dodavatel, odběratel, objednávka a faktura. V daném oddělení má velkým význam rozlišit role pro vedoucího pracovníka a jeho podřízený. Ve vyjmenovaných tabulkách může jenom vedoucí účtárně zpracovávat data pomoci insert a delete. Dané oddělení zpracovává citlivé údaje o dodavatelích konkrétně o jejich bankovních účtech. Podřízení pracovníci účtárně můžou data číst a měnit je.

V případě, že vedoucí účtárně má dovolenou, zpracovávají požadavky na insert a delete pracovníci oddělení procesní podpory. Vedoucí bude mít roli ucetV a podřízení ucetA.

```
CREATE ROLE ucetV;
GRANT ucetV TO zatopek;
GRANT ALL ON dodavatel, faktura TO ucetV;
GRANT select, update ON objednavka TO ucetV;
GRANT select ON zamestnanec, pozice, oddeleni TO ucetV;
CREATE ROLE ucetA;
GRANT ucetA TO, pokorny, mala;
GRANT select, update ON dodavatel, objednavka, faktura TO ucetA;
GRANT select ON zamestnanec, pozice, oddeleni TO ucetA;
```

- **Personální oddělení**

Toto oddělení pracuje v databázi jenom s daty o zaměstnancích. Do jiných tabulek nepotřebují mít přístup ani pro čtení. Je nutní založit znova dvě role. Jedna role bude pro vedoucí a bude se jmenovat personV a druhá bude pro podřízené a bude se jmenovat personA. Je to z toho důvodu, že jenom vedoucí může po ukončení pracovního poměru vymazat data o zaměstnanci z databáze. V její nepřítomnosti se pracovníci oddělení obrátí s požadavkem přímo na IT. Do tabulky zaměstnanci nemají přístup z oddělení procesní podpory ani ke čtení. Je to hlavně z důvodu, že daná tabulka obsahuje citlivá data o platech a tyto údaje jsou zcela osobní, proto vyžadují minimální možný práva k zobrazení.

```
CREATE ROLE personV;
GRANT personV TO polakova;
GRANT ALL ON zamestnanec, oddeleni, pozice TO personV;

CREATE ROLE personA;
GRANT personA TO, pokorny, mala;
GRANT select, update, insert ON zamestnanec, oddeleni, pozice TO personA;
```

- **Procesní podpora**

Procesní podpora je oddělení, které může zpracovávat téměř všechny data v databázi. Tomuto oddělení není umožněno pracovat s tabulkou zaměstnanci. Je to hlavně kvůli informaci, kterou tato tabulka obsahuje a to je plat. Zaměstnanci tohoto oddělení jsou rozděleni na dvě části. Jedna pracuje v databázi a druhá řeší školení a připravuje školící materiály. Dané oddělení má potřebné know-how k mnohým operacím v databázi ale nedisponuje potřebnými informacemi, jinak by mohlo zpracovávat data samo. Smyslem zavedení nové databáze a její zabezpečení není změna struktury v společnosti.

```
CREATE ROLE podpora;
GRANT podpora TO, drobny, novakova;
GRANT ALL ON oddeleni, pozice, artikl, dodavatel, objednavka,
faktura TO podpora;
```

- **IT oddělení**

Toto je poslední oddělení, kterému se vytvoří role. Oddělení IT je také rozděleno na několik pododdělení. Není nutno tedy roli přidělit každému pracovníkovi IT. Toto oddělení má přístup do všech tabulek a může v nich zpracovat vše co je zapotřebí. Pracovníci ale potřebují požadavek k tomu, aby přistoupili k databázi a něco v ní zpracovali. Všechny požadavky se ukládají na intranetu. Není možné, aby pracovník IT vykonal nějaký úkon v databázi bez požadavku z oddělení, které potřebuje zpracovat data v databázi a nemá k tomu dostatečné oprávnění.

```
CREATE ROLE spravce;
GRANT spravce TO, ptak, kysely;
GRANT ALL ON zamestnanec, oddeleni, pozice, artikl,
dodavatel, objednavka, faktura TO spravce;
```

5.2.4. Pohledy

Pohled je zobrazení poskládané z více tabulek do jedné. Data v pohledu jsou jenom ke čtení. Uživatel k nim nemůže přistoupit a změnit je. To je hlavní důvod vzniku pohledů pro mnoho uživatelů, nebo pro celé oddělení, jde hlavně o zamezení přístupu k samotným datům, jejich změně, popřípadě úplné ztrátě.

- **Marketing**

Pohledy v modelové společnosti ke své práci vyžadují mimo jiné pracovníci oddělení Marketingu. Pomocí pohledu zpracovávají další data potřebná k jejich pracovní náplni a pomocí pohledem získaných dat připravují například zboží na akci. Potřebují zobrazit informace o artiklech, jejich prodejní ceně, kdo je dodavatel a kontakt na něj, dále faktury, v kterých se dané zboží objevilo, kvůli přehledu prodeje.

```
CREATE OR REPLACE VIEW Akce AS
SELECT ID_artikl, nazev_artikl, cena_PC, ID_dod, nazev_dod,
tel_c, ID_fa,
FROM Artikl, Dodavatel, Faktura
WHERE artikl.ID_artikl= faktura.ID_artikl
AND artikl.ID_ dodo = dodavatel.ID_dod;
```

Další pohled pro oddělení Marketingu by byl pohled, který by zobrazil údaje o artiklech, dodavatelích, odběratelích a fakturách. Daný pohled by zobrazil prodej na jednotlivých prodejnách. Tento pohled by sloužil k zobrazení prodejnosti artiklů na prodejně k následnému dalšímu využití pro akce cílené na dané prodejně.
Jednou z možností využití daného pohledu oddělením Marketingu by byla oslava samotné prodejny k jejímu výročí a zobrazení úspěšných artiklů v akci pro danou prodejnu.

```
CREATE OR REPLACE VIEW Prodejna AS
SELECT ID_odb, nazev_odb, ID_artikl, nazev_artikl, cena_PC,
ID_dod, nazev_dod, tel_c, ID_fa,
FROM Odberatel, Artikl, Dodavatel, Faktura
```

```
WHERE odberatel.ID_odb=artikl.ID_odb
AND artikl.ID_artikl= faktura.ID_artikl
AND artikl.ID_ dodo = dodavatel.ID_dod;
```

- **Fakturační oddělení**

Další pohled je vytvořen pro fakturační oddělení. Pracovníci sice mají přístup do tabulek zaměstnanec, oddělení a pozice ale museli by získat potřebné informace z tabulek pro ně komplikovanými dotazy. Pohled zobrazuje informaci o zaměstnanci, jeho ID a jméně, dále pak ID oddělení, v kterém pracuje a jméno toho oddělení, a nakonec informace o pozici a jejím názvu. Daný pohled využívá také personální oddělení, a to k rychlému zjištění údajích o zaměstnancích. Údaje, které jsou potřebné k jubilejnímu výročí zaměstnance ve společnosti.

```
CREATE OR REPLACE VIEW zam AS
SELECT ID_osoba, jmeno, prijmeni, ID_odd, nazev_odd,
ID_pozice, nazev_pozice
FROM zamestnanec, oddeleni, pozice
WHERE oddeleni.ID_odd=zamestnanec.ID_odd
AND pozice.ID_pozice=zamestnanec.ID_pozice;
```

- **Vedení společnosti**

Vedení společnosti, jak již bylo zmíněno na začátku předkládané práce, přístup do databáze nepotřebuje. Vedení potřebuje jenom výstupy z ní, a to buď formou reportu, nebo pohledu. Pro její potřeby vzniká pohled, který zobrazuje informaci o zaměstnanci, jeho dodavatelích a artiklech. Dále pak o fakturách, které pracovník z oddělení nákupu umožnil. Tento pohled slouží k přehledu vytíženosti vedoucích pracovníků z oddělení nákupu a k jejich možným odměnám za zisk, který společnosti přinesli.

```
CREATE OR REPLACE VIEW ved AS
SELECT ID_osoba, jmeno, prijmeni, ID_odd, nazev_odd,
ID_pozice, nazev_pozice, ID_dod, nazev_dod, ID_artikl,
nazev_artikl, ID_fa
FROM zamestnanec, oddeleni, pozice, artikl, faktura
WHERE oddeleni.ID_odd=zamestnanec.ID_odd
AND pozice.ID_pozice=zamestnanec.ID_pozice
AND artikl.ID_artikl=faktura.ID_artikl
AND ID_zam=ID_osoba;
```

Výše zmíněný pohled Prodejna, který je vytvořen pro oddělení Marketing využívá také vedení společnosti. Pomocí daného pohledu se také dá zobrazit úspěšnost dané prodejny, kde je vidět prodejna, její artikly a od jakých dodavatelů objednává.

- **Nákupní oddělení**

Pohled pro oddělení nákupu je zobrazení artiklů na jednotlivých prodejnách. Pohled slouží k přesnému zobrazení zařazeních artiklech na jednotlivých prodejnách s dalším zobrazením údajích o nákupčím. Nákupčí food si může zobrazit jenom data o food, pod jeho přihlášením je uložena informace o druhu zboží, který spravuje a toto zboží mu je následně zobrazeno v pohledu.

```
CREATE OR REPLACE VIEW prod AS
SELECT ID_odb, nazev_odb, ID_artikl, nazev_artikl, cena_NC,
ID_odd, nazev_odd, ID_pozice, nazev_pozice
FROM odberatel, zamestnanec, artikl, oddeleni, pozice
WHERE oddeleni.ID_odd=zamestnanec.ID_odd
AND pozice.ID_pozice=zamestnanec.ID_pozice
AND artikl.ID_zam=zamestnanec.ID_osoba
AND artikl.ID_odb=odberatel.ID_odb;
```

5.3. Testování zabezpečení databáze

Postup při testování musí být jednoznačný a pro uživatele snadno pochopitelný. Pro jednotlivá testování musí existovat testovací protokol, do kterého se zaznamenává postup daného testování a jeho výsledek.

Testování probíhá na straně uživatelů, tzn., že každý uživatel databáze se přihlásí svým jménem a heslem. Pak začne pracovat s tabulkami a s daty tak jak to vyžaduje jeho pracovní náplň. Tímto způsobem se odhalí vždy největší počet chyb v nastavení zabezpečení a také se zjistí, že uživatel při prvotním popisu jeho práce na některé své potřeby zapomněl.

Dané testování se provádí vždy po ukončení jedné určité kapitoly projektu. Testování po částech je méně pracné pro změnu v databázi a zároveň se to může znova otestovat.
Výstup z testování, musí být uživatel, který se umí do databáze přihlásit a bezproblémově v databázi zpracovávat své požadavky podle svého oprávnění.

Dále je také nutné otestovat, či se uživatel nedostane k datům, které jsou citlivá a pro jeho pracovní zařazení zcela irelevantní. Také je nutné otestovat, či vytvořená databáze splňuje metody relačně databázové technologie, jako jsou například datová normalizace.

Vývojář, který zpracovává jednotlivé požadavky, testuje a zároveň kontroluje stav zabezpečení v databázi pomocí příkazu v ní.

5.3.1. Systémové pohledy

Pomocí systémového pohledu DBA_TAB_PRIVS se můžou zjistit již oprávnění v databázi.

Describe DBA_TAB_PRIVS

Tabulka 15 Systémový pohled DBA_TAB_PRIVS – vlastní tvorba

Name	Null?	Type
GRANTEE	NOT NULL	VARCHAR2(30)
OWNER	NOT NULL	VARCHAR2(30)
TABLE_NAME	NOT NULL	VARCHAR2(30)
GRANTOR	NOT NULL	VARCHAR2(30)
PRIVILEGE	NOT NULL	VARCHAR2(40)
GRANTABLE		VARCHAR2(3)
HIERARCHY		VARCHAR2(3)

V daném případě nás zajímá, jaká práva byla přiřazena uživateli, dále kdo je vlastníkem objektu.

SELECT grantee, owner, table_name, privilege
FROM DBA_TAB_PRIVS
WHERE grantee=‘novak‘;

Tabulka 16 Zobrazení práv – vlastní tvorba

GRANTEE	OWNER	TABLE_NAME	PRIVILEGE
novak	admin	artikl	SELECT
novak	admin	artikl	UPDATE
novak	admin	artikl	INSERT
novak	admin	artikl	DELETE

Zobrazení nám potvrzuje naše založení práv uživateli Novák z oddělení nákupu na tabulku artikl.

Pomocí dalšího systémového pohledu USER_ROLE_PRIVS se může zjistit seznam rolí přidělených uživateli.

Describe USER_ROLE_PRIVS

Tabulka 17 Systémový pohled USER_ROLE_PRIVS – vlastní tvorba

Name	Null?	Type
USERNAME		VARCHAR2(30)
GRANTED_ROLE		VARCHAR2(30)
ADMIN_OPTION		VARCHAR2(3)
DEFAULT_ROLE		VARCHAR2(3)
OS_GRANTED		VARCHAR2(3)

V daném případě nás zajímá, jaká role byla přiřazena uživateli

SELECT * FROM USER_ROLE_PRIVS

where username=‘novak‘;

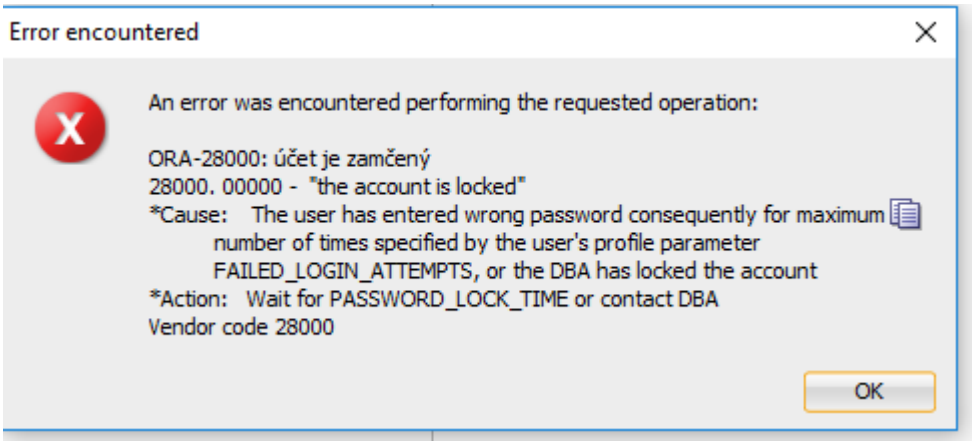
Tabulka 18 Zobrazení role – vlastní tvorba

USERNAME	GRANTED_ROLE	ADMIN_OPTION	DEFAULT_ROLE	OS_GRANTED
novak	NAKUP	NO	YES	NO

Zobrazení nám potvrzuje naše založení role uživateli Novák z oddělení nákupu na roli nakup.

5.3.2. Chybové hlášky

Dalším testem je kontrola funkčnosti uživatelského hesla. Testováním se musí prokázat, že uživatel se nedostane do databáze, když se přihlásí opakovaně špatným heslem, a to zablokováním účtu.

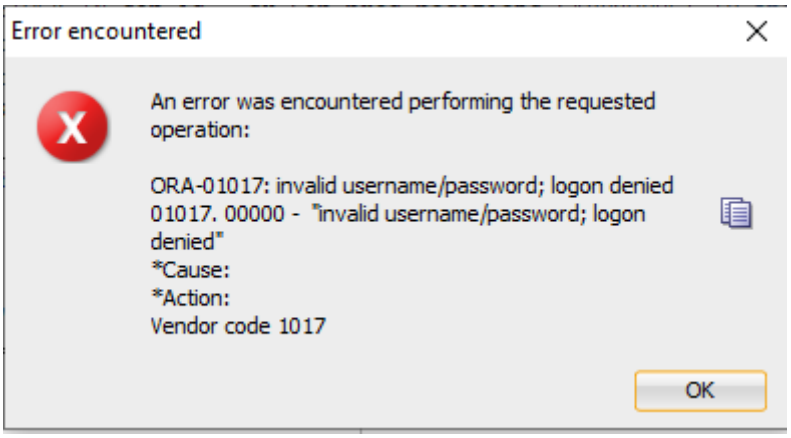


Obrázek 7 Chybová hláška – účet je zamčený – vlastní tvorba

Chybová hláška nám oznamuje, že se uživatel přihlašoval špatným heslem. Toto heslo zadal špatně maximálně krát tak jak je nastaveno v FAILED LOGIN ATTEMPTS, což v našem případě jsou 3 pokusy. Dále nám hlásí, že uživatel musí počkat dobu, která je nastavená v parametru PASSWORD_LOCK_TIME, a to je v našem případě jedna hodina. Po uplynutí této doby se bude moct uživatel znova přihlásit.

Dané parametry, které jsou nastaveny na limity, se mohou změnit, jestliže nevyhovují vnitřní politice zabezpečení databáze. Je ale nutní mít na paměti, že uživatelé jsou kreativní a může se stát, že uživatel i náhodou přídě na to jak dané opatření obejít. Proto samotnému nastavení je nutno věnovat značnou pozornost.

Chybová hláška na obrázku 8 informuje uživatele, že bylo použito nekorektní uživatelské jméno nebo heslo.



Obrázek 8 Chybová hláška – nekorektní uživatelské jméno nebo heslo – vlastní tvorba

Nezbytným testem je také pokus o spuštění operace viz. obrázek 9, ke které nemá uživatel přístup a daná hláška je zobrazená v takovém případu správně. Jestliže by dané nastavení oprávnění nebylo dostačující a uživatel by měl mít k operaci přístup, tak správním testováním se tyto nedostatky odhalí. Proto je nutný zapisovat jednotlivá zjištění do protokolu. Po ukončení testování se daný protokol vyhodnotí a následně se zapracují nedostatky.

```
Error report -
SQL Error: ORA-01031: insufficient privileges
01031. 00000 -  "insufficient privileges"
*Cause:    An attempt was made to perform a database operation without
           the necessary privileges.
*Action:   Ask your database administrator or designated security
           administrator to grant you the necessary privileges
```

Obrázek 9 Chybová hláška – nedostateční oprávnění – vlastní tvorba

6. Diskuse

Žijeme v moderní době, a proto se nemůžeme divit, že obsah dat, který se zpracovává v databázích, rychle narůstá. Tím vzniká také otázka, jak tyto data schraňovat, obhospodařovat, ukládat a také zabezpečit. Toto je otázka, se kterou se potýkají všechny společnosti, které pracují s daty. A mohlo by se říct, že je to výzva pro mnoho společností, zvítězit nad tím, jak správně zabezpečit svá data. Řešení existuje určitě spousta, a předkládaná diplomová práce se věnuje jednomu z možných návrhů zabezpečení přístupu k datům vevnitř společnosti.

Na samotném začátku předkládaného řešení se věnovala pozornost podle teoretických znalostí návrhu zabezpečení databáze. V modelové společnosti byl jeden velice podstatný problém a to ten, že zaměstnanci mnoho krát obcházeli databázi, která v společnosti existovala, a zaznamenávali svá data mimo ni.
Proto již na samém začátku práce se zabezpečením databáze se musí společnost jako celek spojit a začat spolupracovat. V reálním světě to znamená, že vedení společnosti musí své zaměstnance řádně proškolit a zavést tyto skutečnosti do interního myšlení ale také do všech směrnic, kterých se tato skutečnost týká. V tomto bodě se může stát, že veškerá práce nad zabezpečením dat bude k ničemu, jestli zaměstnanci nebudou řádně spolupracovat a před ztrátou zaměstnání kvůli neplněním svých povinností znehodnotí nebo poškodí data v databázi. V tomto bodě je možný problém, který je potřeba řádně podnikem promyslet. Proto je nutné jako řešení problému ještě před samotním zabezpečením databáze namotivovat uživatelé k práci v novém systému. Motivace musí být přímá a musí být vidět, že novým systémem si uživatel polepší. Musí být jasně ukázáno, že nový systém bude jednoduchý na obsluhu. Ze strany zaměstnavatele se musí uživatelé ubezpečit, že se začne pracovat s novým systém až po jeho kompletním zavedení do společnosti, kdy nový systém bude plně funkční bez zásahu dodavatelské společnosti.

Dané společnosti byl navrhnut nový systém, který byl zobrazen jen okrajově a naplněn daty. Při tvorbě databáze se dodrželi všechna pravidla tak aby byly definovány relační vztahy mezi nimi jednotlivými tabulkami.

Dále se pracovalo na normalizaci databáze co je proces, při němž dochází k redukci opakování a nadbytečnosti dat v databázi a na integritě dat. Toto jsou důležité kroky k zavedení databáze do života. Bez správně fungující databáze není důležité přejít k jejímu zabezpečení.

Další důležitou částí byl popis jednotlivých pracovních pozic, které do systému budou mít přístup. Je velice nutné přesně identifikovat uživatele, jejich práva, povinnosti a možnosti. Na tomto základě byly podrobně projety uživatelské účty, profil, přístupové role, práva a možné pohledy.

Po samotném nastavení zabezpečení databáze jsme se dostali do poslední fáze a to testování. Testování probíhalo v posloupnosti jednoduchých testů, které by v případě problému odhalilo chybu nebo naopak zobrazilo správné zabezpečení přístupu do databáze.

Navrhnuté zabezpečení databáze v praktické části této předkládané diplomové práce má sloužit jako ukázka možného řešení v dané problematice, které by se dalo využít v malém podnikatelském subjektu. Dalšími možnými zlepšeními v zabezpečení databáze by bylo například šifrování, zálohování nebo také audit. To ale samotnou kvalitu ukázky zabezpečení databáze v praktické části nedegraduje a je plně využitelná.

7. Závěr

Diplomová práce byla zaměřená na problematiku zabezpečení provozu databázově koncipovaných informačních zabezpečení malých a středních podnikatelských subjektů. Hlavním cílem této předkládané diplomové práce bylo zabezpečení databázové evidence v malém podnikatelském subjektu, které vedlo ke snížení neoprávněného přístupu vevnitř společnosti, a to konkrétně do samotné databáze, dále byl omezený přístup uživatelům k samotným datům a práce s nimi. Pro tyto účely byla vytvořená fiktivní databáze s daty v rozsahu dostačující pro zabezpečení přístup a také uživatelé.

Stěžejními metodami předkládané práce byly metody relačně databázové technologie, jako jsou datové modelování, datová normalizace, integrita dat a techniky přidělování práv přístupu. Principy zabezpečení databáze, mezi které patří přehled dat v databázi a možný přístup a práce s nimi. Dále pak nastavení pravidel přístupu do databáze, práce s uživatelskými účty, hesly, tvorba rolí, profilů a pohledů.

Dílčími stěžejními kroky daného zabezpečení bylo zmapování současného stavu a zjištění požadavků na nový systém dále pak návrh datového modelu a požadavky na přístup do databáze. Na základě výše popsaném postupu se realizovalo zabezpečení a testování přístupu do databáze, přístupu k datům a práce s nimi.

K naplnění vytýčených cílů byla použita studia a analýza dostupných informačních zdrojů a existujících řešení. Stěžejními metodami předkládané práce byly především metody a techniky relačně databázové technologie využívané v kontextu s problematikou zabezpečení databáze.

Diplomová práce vysvětluje možnost zabezpečení databáze před neoprávněným zásahem ze strany koncových uživatelů vevnitř dané společnosti. Také poukazuje na to, že dané řešení zabezpečení databáze je nutné, a přínosem je kontrolovatelný přístup do databáze. Popisovaný přínos byl realizován na modelové databázi, a to omezením přístupu do databáze uživatelům, který daný přístup potřebují k vykonávání své pracovní náplně. Dané omezení bylo zpracováno tak aby byl zajištěn omezený přístup k citlivým datům.

Podnikatelské subjekty pracují s velice citlivými daty, jako jsou data o zaměstnancích, o dodavatelích a v neposlední řade samotná data o výrobcích neboli zboží. Tyto data jsou pro konkurenci velice atraktivní, proto je nutné přístup k nim zabezpečit. Prvním krokem proto bylo tyto citlivé údaje identifikovat. K tomu bylo zapotřebí porozumět logice dat v společnosti.

K naplnění cílů bylo nutné nastudovat relačně databázové technologie a možnosti Oracle. Postup při zabezpečení databáze byl dán studiem v dané problematice. Jednotlivé kroky byly ale logické a postupně navazovali na sebe. Po podrobné analýze informačních zdrojů se práce posunula do tvorby teoretické části diplomové práce.

Po tvorbě teoretické části se zmapoval momentální stav řešené problematiky. Zmapování bylo jenom na teoretické bázi. Nebylo možné zmapovat momentální stav v konkrétní databázi. Tato skutečnost ale umožnila další vzdělávání v daný oblasti, a to i na internetových stránkách, které se daný problematice věnují již delší dobu.

Samotný návrh zabezpečení databáze vycházel z nastudování informačních zdrojů. Bylo nutné se zamyslet také nad tím, že i v dnešní době malé podniky spoléhají na loajálnost svých zaměstnanců. Co není špatný, ale v businessu to nestačí. Proto je nutné přístup do databáze uživatelům omezit na nejnutnější.

V popisu modelové společnosti bylo zapotřebí zjištění současného stavu, a to vedlo k tvorbě datového modelu. Návrh zabezpečení databáze vycházel také z požadavků samotné společnosti a samozřejmě také z požadavků uživatelů. Jednotlivé oddělení v modelové společnosti, byly řádně popsány, a daný popis byl následně sepsán do konkrétní podoby.

Samotné nastavení profilu se zpracovalo tak aby vyhovovalo zabezpečení databáze a nebylo možné se přihlásit po neúspěšném pokusu. Dané parametry, které byly nastaveny by se mohly změnit, jestliže by nevyhovovali vnitřní politice zabezpečení databáze.

Bylo ale nutné mít na paměti, že uživatelé jsou kreativní a mohlo by se stát, že uživatel i náhodou přijde na to, jak dané opatření obejít. Proto bylo samotnému nastavení nutné věnovat značnou pozornost.

Pro nastavení role uživatelům se zpracovala tabulka, která popisovala potřebné přístupy uživatelům do databáze. Daná tabulka byla přehledná a nastavení role z ní bylo velice jednoduchý. Z tabulky byly údaje pro tvorbu role lehce čitelné, a proto byla tato tabulka přínosem při tvorbě rolí v dané malé modelové společnosti. Omezení dané tabulky by bylo ve velké společnosti, kde by mohlo dojít k nepřehlednosti těchto údajů.

Po samotné tvorbě uživatelských účtů, rolích profilů a pohledů, bylo velice důležité samotné testování. Při testování se simulovali reální situace a výstupem bylo zjištění, že nastavená zabezpečení jsou správná a odpovídají na předem domluveném přístupu. Dané testování se provádělo vždy po ukončení jedné určité kapitoly projektu. Testování po částech je méně pracné pro změnu v databázi a zároveň se to může znova otestovat.
Výstupem z testování, bylo to, že konkrétní uživatel, se uměl do databáze přihlásit a bezproblémově v databázi zpracovávat své požadavky podle svého oprávnění a pracovního zařazení. Dále bylo také nutné otestovat, jestli se uživatel nedostane k datům, které jsou citlivá a pro jeho pracovní zařazení zcela irelevantní.

Relační databáze se stále zdokonalují, proto zavedení a vývoj databáze v společnosti je jenom kladným přínosem. Je ale nutno k dané problematice přistupovat zodpovědně aby nedošlo k již zmiňované ztrátě dat. Dalo by se to pojmenovat jako cíl, který by měla společnost zavést a ukotvit do interních směrnic.

Seznam použitých zdrojů

1. STEPHENS, Ryan K. a Ronald R. PLEW. *Naučte se SQL za 21 dní: pochopte principy jazyka relačních databází: uplatněte získané dovednosti při tvorbě dotazů a databázových aplikací*. Brno: Computer Press, 2004. ISBN 8072268708.
2. STEPHENS, Ryan K., Ronald R. PLEW a Arie JONES. *Naučte se SQL za 28 dní: [stačí hodina denně]*. Brno: Computer Press, 2010. ISBN 978-80-251-2700-1.
3. GROFF, James R. a Paul N. WEINBERG. *SQL: kompletní průvodce*. 2005. Brno: CP Books, 2005. Programování. ISBN 8025103692.
4. VALENTA, M., POKORNÝ, J. *Databázové systémy*. Praha: České vysoké učení technické v Praze, 2013. ISBN 978-80-01-05212-9.
5. WALTERS, R. E. *Mistrovství v Microsoft SQL Server 2008: [kompletní průvodce databázového experta]*. Brno: Computer Press, 2009. ISBN 9788025123294.
6. HERNANDEZ, Michael J. *Návrh databází*. GRADA, 2006. ISBN: 80-247-0900-7
7. PROCHÁZKA, David. *Oracle: průvodce správou, využitím a programováním nad databázovým systémem*. Praha: Grada, 2009. Průvodce (Grada). ISBN 978-80-247-2762-2.
8. LONEY, Kevin. *Oracle Database: kompletní průvodce*. Brno: Computer Press, 2010. Administrace (Computer Press). ISBN 9788025124895.
9. LACKO, Ľuboslav. *Databáze: datové sklady, OLAP a dolování dat s příklady v Microsoft SQL Serveru a Oracle*. Brno: Computer Press, 2003. ISBN 8072269690.
10. THERIAULT, Marlene a Aaron NEWMAN. *Bezpečnost v Oracle: metody, nástroje a řešení problémů: [platné pro Oracle 9i, 8i, 8 a 7.x]*. Brno: Computer Press, 2004. Security (CP Books). ISBN 80-7226-979-8.
11. BASL, Josef a Roman BLAŽÍČEK. *Podnikové informační systémy: podnik v informační společnosti*. 3., aktualiz. a dopl. vyd. Praha: Grada, 2012. Management v informační společnosti. ISBN 9788024743073.
12. BRYLA, Bob a Kevin LONEY. *Mistrovství v Oracle Database 11g*. Brno: Computer Press, 2009. ISBN 9788025121894.

13. 5 tipů, jak zabezpečit databázi - Dvojklik. *Dvojklik - Magazín o informační bezpečnosti* [online]. Copyright © 2019 [cit. 25.11.2019]. Dostupné z: https://www.dvojklik.cz/5-tipu-jak-zabezpecit-databazi/
14. Úvod do systému uživatelských práv MySQL. *Programujte.com — odborný web zaměřený na oblast vývoje, návrhu a designu webových, mobilních a desktopových aplikací* [online]. Copyright © 2003 [cit. 28.11.2019]. Dostupné z: http://programujte.com/clanek/2008090900-uvod-do-systemu-uzivatelskych-prav-mysql/
15. Na co si dát pozor při návrhu databáze? - Root.cz. *Root.cz - informace nejen ze světa Linuxu* [online]. Copyright © 1998 [cit. 28.11.2019]. Dostupné z: https://www.root.cz/clanky/na-co-si-dat-pozor-pri-navrhu-databaze/
16. Myslíte si, že mají databázoví specialisté nudnou práci? Tomáš Solař vám odpoví, že ne, i proč. Programujte.com — odborný web zaměřený na oblast vývoje, návrhu a designu webových, mobilních a desktopových aplikací [online]. Copyright © 2003 [cit. 26.02.2021]. Dostupné z: http://programujte.com/clanek/2016010801-myslite-si-ze-maji-databazovi-specialiste-nudnou-praci-tomas-solar-vam-odpovi-ze-ne-i-proc/
17. LACKO, Ľuboslav. SQL: hotová řešení. Brno: Computer Press, 2003. K okamžitému použití (Computer Press). ISBN 80-7226-975-5.
18. LACKO, Ľuboslav. *Oracle: správa, programování a použití databázového systému*. 2., dopl. vyd. Přeložil Marek KOCAN. Brno: Computer Press, 2007. ISBN 978-80-251-1490-2.