Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Diplomová práce

Deduplikace klientských dat

Marek Chytrý

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Marek Chytrý

Informatika

Název práce

Deduplikace klientských dat

Název anglicky

Deduplication client data

Cíle práce

Diplomová práce je tematicky zaměřena na problematiku čištění klientských dat. V teoretické části budou popsány metody určení čistoty dat, jejich vyhodnocení a zpracování, v praktické části pak ukázka jednoho ze způsobu čištění deduplikace klientských dat.

Metodika

V teoretické části budou popsány metodiky konsolidace dat. Znalosti získané v teoretické části budou prakticky využity při implementaci logiky čištění dat s využitím deduplikace. Výstupy z praktické části budou ověřeny pomocí testovacích scénářů. V závěru pak budou shrnuty poznatky z teoretické a praktické části a vyhodnocení výsledků.

Doporučený rozsah práce

40 – 60 stran

Klíčová slova

deduplikace dat, kvalita dat, index čistoty, koeficient čistoty, data, databáze

---

Doporučené zdroje informací

HE, Qinlu; LI, Zhanhuai; ZHANG, Xiao. Data deduplication techniques. In: Future Information Technology and Management Engineering (FITME), 2010 International Conference on. IEEE, 2010. p. 430-433.

Chapman, A. D. 2005. Principles and Methods of Data Cleaning Primary Species and Species Occurrence Data, version 1.0. Report for the Global Biodiversity Information Facility, Copenhagen.

MANDAGERE, Nagapramod, et al. Demystifying data deduplication. In: Proceedings of the ACM/IFIP/USENIX Middleware'08 Conference Companion. ACM, 2008. p. 12-17.

Oracle(r) Warehouse Builder Data Modeling, ETL, and Data Quality Guide

Robert Laberge. Datové sklady: agilní metody a business intelligence.1. vyd. – Brno : Computer Press, 2012. – 350 s.

VANÍČEK, Jiří. Měření a hodnocení jakosti informačních systémů. Česká zemědělská univerzita, Provozně ekonomická fakulta, 2004.

11g Release 2 (11.2) [online]: Redwood City 21.9.2011. (http://docs.oracle.com/cd/E11882_01/owb.112/e10935/toc.htm)

Předběžný termín obhajoby

2016/17 ZS – PEF

Vedoucí práce

Ing. Jan Tyrychtr, Ph.D.

Garantující pracoviště

Katedra informačních technologií

Elektronicky schváleno dne 31. 10. 2014

Ing. Jiří Vaněk, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 11. 11. 2014

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 20. 11. 2016

Čestné prohlášení

Prohlašuji, že svou diplomovou práci "Deduplikace klientských dat" jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu literatury na konci práce. Jako autor uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 30.11.2016 _____

Poděkování

Rád bych touto cestou poděkoval Ing. Janu Tyrychtrovi, Ph.D. za odborné vedení a pomoc při zpracování diplomové práce. Dále bych chtěl poděkovat své manželce a rodičům za oporu, povzbuzení a vytváření příjemného prostředí pro psaní této práce.

Deduplikace klientských dat

Deduplication client data

Souhrn

Práce se zabývá problematikou kvality a čištění dat. V teoretické části jsou nejprve ukázány vlastnosti kvality dat a jsou zde naznačeny oblasti, kterých se tyto vlastnosti v IT i mimo ně týkají. Následně jsou popisovány možnosti ukládání dat v informačních systémech opět z pohledu kvality a její možné dopady na efektivnost užití těchto dat. V další části jsou charakterizovány různé přístupy ke kvalitě dat z úrovně managementu, při pohledu na celou společnost atd. V neposlední řadě jsou zmiňovány další možnosti řízení kvality dat přes různé standardy, normy atd.

Druhá část teoretické práce se věnuje tématu čištění dat. V rámci této části jsou definovány různé druhy chyb a špatných záznamů, které chceme odstranit / opravit. Je zde popsán postup čištění dat, který je nutné provést, aby čištění dat bylo co nejefektivnější.

V praktické části je pak popsána implementace konkrétní metody čištění dat, a to deduplikace. Praktická část obsahuje nejen skript pro deduplikaci, ale i další přípravné skripty a pomocné čistící skripty, které připravují data pro čištění i samotnou deduplikaci.

V závěru práce je vyhodnocena úspěšnost jak čistících skriptů, tak deduplikačního skriptu.

Klíčová slova

kvalita, kvalita dat, klient, databáze, čištění, deduplikace dat

Summary

The thesis includes quality and data cleansng issues. Firstly, characteristics of data quality are described in the theoretical part. There are also indicated some areas not only in IT that these characteristics relate to. Further, there are also described the possibilities of storing data in information systems again in terms of quality and their potential impacts. The next section shows different approaches to data quality from the management view, company view etc. There are also mentioned other possibilities of data quality management through a variety of standards, norms etc.

The second part of the theoretical work focuses on data cleaning issue. In this part there are defined different types of errors and bad records that we want to remove / repair. There is also described a necessary process of data cleaning that enables us to get as efficient data cleaning as possible.

The practical part describes the implementation of a particular method of data cleaning, i.e. deduplication. The practical part includes not only the script for deduplication but other preparatory and auxiliary cleaning scripts as well. These scripts prepare data for cleaning and deduplication.

In the section of conclusion, the cleaning and the deduplication scripts are evaluated.

Keywords

quality, data quality, client, database, cleaning, data deduplication

Obsah

1 Úvod

Zaznamenávání dat do informačních systémů je hlavním důvodem, proč se tyto systémy používají. Pokud by ale tato data neměla žádnou míru informace pro uživatele, pak by pro nás zřejmě nebyla zajímavá. Dle Shannonovy teorie by pak snižovala požadovanou míru určitosti a byla by pro nás neužitečná (Vaníček, 2007). Proto, aby informační systémy měly co největší přidanou hodnotu, musí obsahovat nejen velké množství informací, ale především kvalitní informace.

Samotná data v databázích nemají žádnou kvalitu nebo hodnotu, obsahují pouze informace. Tyto informace mají pouze potenciální kvalitu, která je využita, když jsou data potřebná nebo použitelná. Tedy výše kvality informací je přímo úměrná schopnosti uspokojit zákazníka nebo jeho potřeby. (Dalcin, 2005)

Další neméně důležitou úlohou informačních systémů je archivace informací, jedná se o takzvané datové sklady. V těchto případech nezvyšujeme míru hodnoty informace, ale chceme daná data uchovat a v případě potřeby opět nalézt. Ale aby tato uložená data byla jednoduše znovupoužitelná, musí být logicky uspořádána a strukturovaná. Jedná se o vlastnosti znovupoužitelnosti, což je jeden z atributů kvality. Těchto atributů je více a postupně budou popsány v této práci.

Kvalita dat je pojem, který bude v této práci velmi často používaný. Co všechno tento pojem skrývá? Velmi jednoduše z pohledu zákazníka lze říci, že co má vyšší kvalitu, je pro zákazníka lepší… Lepší, ale v jakém ohledu? Někdy je kvalita čistě subjektivní. Podle Chapmena „Principy kvality dat“ (Chapman, 2005) má pojem „kvalita dat“ více definic.

Nejčastěji se používá pojem „kvalita“ jako synonymum pro „Vhodnost použití“ nebo „Potenciální použití“. Podle profesora Vaníčka je kvalita definována jako stupeň uspokojení daných nebo stanovených požadavků zainteresovaných stran souborem vnitřních vlastností posuzovaného objektu (Vaníček, 2010). V obou zmíněných definicích se opakuje pojem použití. A použití respektive využití je důležité nejen v informačních technologiích, ale je i součástí kvality našich každodenních životů.

Z manažerského pohledu je požadavek velmi jednoduchý: „Uložené informace by měly v maximální míře podporovat dané podnikání“, tj. jak bylo v definicích kvality zmíněno výše: „Měly by být maximálně využité“. Tato definice se zdá být velmi jednoduchá, ale

realizace je složitější. Nejen, že data musí být přístupná 24/7 (denně dvacet čtyři hodin, sedm dní v týdnu), musí být dostupná pro daný počet uživatelů, musí být perzistentně uložena, musí být aktuální i kompletní atd. Navíc u velkého množství dat není zcela jednoduché zajistit, aby uchovávaná data a hlavně systémy, které s nimi pracují, splňovaly všechny požadavky, které na ně uživatel (manažer) klade.

Aby podpora kvality informací byla zohledňována celým podnikem, je ji potřeba zakotvit už ve vizích organizace. Správně definovat a stanovit priority jednotlivých požadavků, přiřadit k nim odpovědné osoby atd. Pokud toto nebude opomenuto, dojde jednodušeji k lepšímu naplňování vize celé společnosti (Redman, 2001) a dojde k zefektivnění procesů v organizaci. (Chapman, 2005)

V rámci úvodu byly zmíněny některé pojmy týkající se této problematiky. Je zřejmé, že se jedná o velmi komplexní oblast IT, která ovlivňuje systémy napříč jednotlivými obory. Víc než kde jinde je důležitá podpora a vzájemné pochopení potřeb, požadavků na IS a následná vzájemná spolupráce celé organizace. V rámci požadavků je velice důležité jednoznačně a detailně specifikovat, jak mají data vypadat a jak mají být přístupná, aby byla pro uživatele použitelná v té míře, v jaké očekává. Jednotlivým oblastem se budou dále věnovat jednotlivé kapitoly, které budou především zaměřeny na problematiku týkající se perzistentních úložišť a na této úrovni ovlivnění kvality dat. Praktická část pak bude obsahovat skript naprogramovaný v procedurálním jazyce relační databáze Oracle a bude sloužit ke zvyšování kvality informace v klientských záznamech.

2 Cíl práce a metodika

2.1 Cíl práce

Cílem práce je vytvořit program, který zvýší kvalitu dat. Tento program obsahuje jeden konkrétní proces čištění – deduplikaci klientských dat. Výsledkem práce pak je nejen samotný program, ale i určité metody měření kvality dat, které ukážou, o kolik se kvalita dat změní.

2.2 Metodika

V rámci teoretické části jsou nejprve definovány pojmy kvality a čištění dat. Dále jsou popsány různé přístupy k samotnému hodnocení kvality dat. Z těchto metodik je jedna vybrána a pomocí ní je vyhodnocena kvalita dat čistícího programu z praktické části práce. V teoretické části jsou dále popsány chyby a potažmo obecně definice nekvalitních dat, které vyplývají z požadavků zadavatelů.

V rámci konkrétní metodiky jsou zpracovány požadavky, které vyhodnotí, zda implementovaný skript v praktické části zlepší deklarovanou kvalitu dat.

Praktická část navazuje na popisovaná nekvalitní data a má za cíl určitý typ nekvality dat odstranit/nahradit. Kvalita dat je následně změřena jak před během čistícího skriptu, tak po něm.

V praktické části při vyhodnocování kvality dat je potřeba postupně připravit hodnocení, poté definovat požadavky, pak specifikovat hodnocení a provést návrh hodnocení a na úplném závěru provést samotné hodnocení.

Pro důvěryhodnější výsledky jsou všechna měření opakována stokrát a následně je těchto sto hodnot zprůměrováno. Ve výsledcích jsou zaznamenány tyto průměry.

Praktická část je implementována v procedurálním jazyce PL-SQL.

2.2.1 Jazyk PL-SQL

PL-SQL neboli procedurální SQL je nástavba databázového dotazovacího jazyka SQL (select query language). Tento jazyk je vyvíjen společností Oracle a je součástí stejnojmenné databáze. Velkou výhodou tohoto programovacího jazyka je, že pracuje přímo s daty z databáze, proto zde nejsou žádné potřebné přenosy dat a díky tomu je

zpracování rychlejší. Další velkou výhodou je využívání jazyka SQL v rámci skriptů. Není proto potřeba žádných dalších „mezi“ dotazovacích jazyků, jako je například jSQL a další.

V rámci tohoto jazyka je možno používat objekty, dědičnosti objektů, procedury/funkce vázané na tyto objekty a další prvky objektového programování. Jedná se plně o objektové programování s podobnými možnostmi, co nabízejí ostatní objektové jazyky.

Úplným základem programování v tomto jazyce je nepojmenovaný blok. Ten se dělí do tří částí:

- V první části jsou deklarovány proměnné. Proměnné jsou zde pojmenovány a jsou jim přiřazeny odpovídající datové typy. Samozřejmě je možné vytvořit vlastní datový typ/nový objekt a ten si uložit. Následně je možné tento typ dále používat.
- V druhé části je výkonná část programovacího skriptu. Zde se programuje to, co má daný skript provádět, co má zpracovávat, popřípadě co má dál předávat/ukládat atd.
- Ve třetí části se odchytávají výjimky a probíhá jejich zpracování. Podle druhu výjimky je možné ji předat dál, nebo ošetřit na daném místě.

První a třetí část je volitelná. V případě, že nechcete nebo nepotřebujete deklarovat proměnné a odchytávat výjimky, je možné skript napsat i bez těchto částí.

Ukázka jednoduchého nepojmenovaného bloku, který vypíše do konzole „Hello World!“

```
--dvojitou pomlčkou jsou označeny komentáře
DECLARE--uvozuje první část
  --deklarace proměnné
  greeting varchar2(20);
BEGIN--uvozuje druhou část
  --vybere z tabulky pozdravů, pozdrav s id = 1
  SELECT g.value
    INTO greeting
    FROM table_greetings g
   WHERE g.id = 1;
   --pozdrav je vypsán do konzole
  DBMS_OUTPUT.PUT_LINE(greeting);
EXCEPTION--uvozuje třetí část
  --definice výjimky, která je odchytávaná
  --je odchytáván případ, kdy id = 1 neexistuje
  WHEN NO_DATA_FOUND THEN
  --uživatel je o daném stavu informován
  DBMS_OUTPUT.PUT_LINE('Nenalezen žádný pozdrav');
END;--konec nepojmenovaného bloku
```

Kód 1: Hello world v PL-SQL

Výpis ve výstupu po spuštění tohoto bloku je následovný: `Hello world!`

Jednotlivé bloky je možno pojmenovávat a vytvářet tak procedury, ty žádnou hodnotu nevracejí, nebo funkce, ty mají navíc deklarovanou návratovou hodnotu. Posléze je možné takto pojmenované procedury a funkce zanořovat do balíčků, a ty také pojmenovávat. Většinou jeden balíček obsahuje ucelenou funkčnost. V mé práci jsou jednotlivé moduly uloženy v jednotlivých balíčcích. Například dirty Modul je uložen v balíčku PCK_MAKE_DIRTY.

Navíc je možné používat předpřipravené funkčnosti od společnosti Oracle. Tyto funkčnosti jsou také sdružovány do balíčků. Jedná se o různé pomocné metody pro zpracovávání textů, matematické operace atd. V rámci mé práce jsem například použil připravenou funkčnost na porovnávání podobnosti textových řetězců. Tato funkčnost je uložena v balíčku UTL_MATCH a díky tomu nebylo potřeba tuto funkčnost implementovat.

3 Teoretická východiska

3.1 Kvalita dat

Kvalita dat, jak bylo zmiňováno výše, souvisí s jednoduchostí užití. V angličtině se používá pojem „fit for use“ a nelze ji posuzovat bez závislosti na uživateli (Chapman, 2005), protože ten je hlavním článkem, který s daty pracuje. Které jsou ty části kvality dat, které mají být ovlivněny, aby se zvýšila efektivnost užití? Obecně podle Chapmena (2005) je vlastností, na které se lze zaměřit, hodně, ale v konečném důsledku je třeba identifikovat konkrétní vlastnosti pro konkrétního uživatele a konkrétní účel užití dat.

Příklad uvádí dopad na kvalitu dat způsobený rozpory mezi zadavateli a IT. V zemědělském podniku je algoritmus pro výpočet potřebného množství hnojiva. Algoritmus ukládá výsledek správně, ale je zde zaznamenána špatná jednotka. Vůbec nezáleží na tom, zda by touto chybou bylo způsobeno hnojení zvýšené nebo snížené o tři řády. Každá z chyb by měla jiný dopad, ale ne nezanedbatelný. Tento ukazatel kvality je označován jako správnost dat.

Jiným příkladem chybného zpracování dat je špatně zvolené zaokrouhlování. Můžeme uvažovat situaci, kdy v zemědělském podniku je sklízeno obilí a jsou zpracovávány údaje o jeho množství. Potřebujeme evidovat množství a vyúčtování pohledávek zákazníkovi. Údaje z vah jsou dostatečně přesné na desetiny kilogramů, ale v datech jsou hodnoty uložené a zaokrouhlené na tuny. V závislosti na tom, jak budou data zaokrouhlována, zda matematicky, vždy nahoru, vždy dolů, bude buď zákazník, nebo náš podnik znevýhodněn.

3.1.1 Vlastnosti kvality dat

Vlastnosti kvality dat můžeme brát jako to, co je potřeba udělat s informačním systémem, aby se zvýšila použitelnost pro širší okruh uživatelů daného systému (tj. zvýšilo se potenciální využití), a díky tomu může být systém použit k více účelům. Na druhou stranu musíme vybalancovat mezi zvýšením použitelnosti a mírou úsilí, které je potřeba vynaložit pro přidání funkcionalit a zvýšení užitečnosti. (Chapman, 2005)

Podle Redmana (2001) se jedná zejména o tyto vlastnosti:

- Přístupnost (Accessible)
- Přesnost (Accurate)
- Preciznost (Precission)

- Včasnost (Timely)
- Kompletnost (Complete)
- Konzistentnost s jinými zdroji (Consistent)
- Relevantnost (Relevant)
- Obsáhlost (Comprehensive)
- Správná úroveň detailu (Proper Level of Detail)
- Pochopitelnost (Easy to Read)
- Jednoduchost interpretace (Easy to Interpret)

Vlastnosti kvality dat jsou množinou, která obsahuje vlastnosti, které mohou mít dopad na použitelnost dat. Pokud nějaká vlastnost ovlivňuje rychlost nebo efektivitu práce uživatele s daty, pak by zde měla být uvedena. Pan profesor Vaníček nazývá vlastnosti kvality dat charakteristikami kvality dat a kromě výše zmíněných uvádí navíc tyto charakteristiky (2010):

- Platnost (Currency)
- Přesnost (Accuracy)
- Zabezpečení (Security)
- Obnovitelnost (Recoverability)
- Ovladatelnost (Manageability)
- Účinnost (Efficiency)
- Proměnlivost (Changeability)
- Přenositelnost (Portability)
- Produktivnost (Productivity)
- Bezpečnost (Safety)
- Důvěryhodnost (Credibility)
- Dostupnost (Availability)
- Shoda s předpisy (Regulatory Compliance)

Charakteristik by bylo možné najít ještě mnohem víc a další autoři definují buď další charakteristiky, nebo ty samé pojmenovávají trochu odlišnými pojmy a popisují je z trochu jiného úhlu pohledu. (McGilvray, 2008)

V konečném důsledku ale záleží především na zákazníkovi, co od dat očekává a k jakému konkrétnímu účelu mají být využita, a na základě toho je potřeba specifikovat požadavky na kvalitu dat.

Podrobnější popis jednotlivých charakteristik kvality dat:

3.1.1.1 Přístupnost (Accessible)

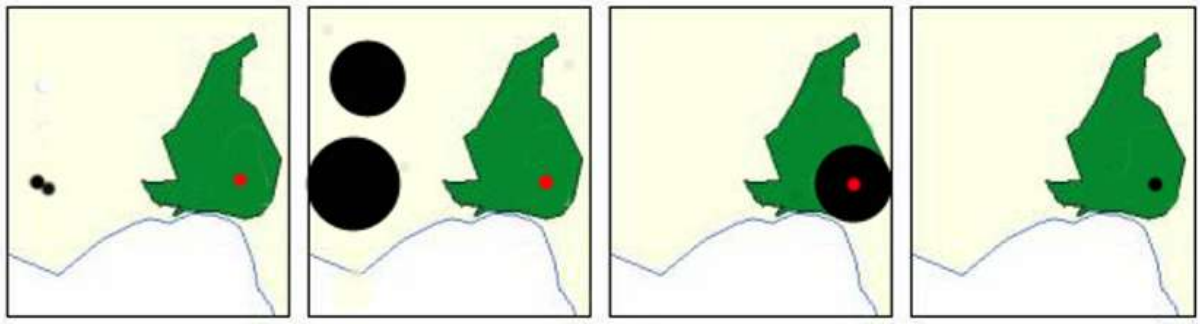
Přístupnost typicky definuje aktivity jako ukládání, načítání dat nebo další aktivity spojené s databázemi nebo jinými typy úložišť. Jsou to postupy, které zabezpečují přístup k datům v místě, kde jsou data reálně fyzicky uložena.

Historicky bylo potřeba implementovat různá rozhraní a jazyky, které byly požadovány na základě úložiště nebo operačního systému. Navíc existovaly různé odlišné nekompatibilní formáty přístupu k datům. Stále rozdíly existují, ale jsou již daleko menší. A jsou zde stále větší snahy o jejich sjednocování. Typickými standardy jsou SQL, ODBC, JDBC a další. Navíc některé standardy umožňují překlad dat z nestrukturované (HTML, free-text) do strukturované formy (XML nebo SQL). (McGilvray, 2008)

3.1.1.2 Přesnost /správnost (Accuracy) a preciznost (Precision)

Přesnost a preciznost jsou dvě vlastnosti kvality dat, které se na první pohled mohou zdát shodné, až zaměnitelné. A pokud budeme chtít výrazy přeložit, v obou případech dostaneme slovo „přesnost“. Jaký je v nich tedy rozdíl? Tyto vlastnosti spolu velmi souvisí a navzájem se doplňují. Jejich vysvětlení je nejjednodušší na konkrétním příkladu, viz grafické znázornění.

Úkol: Určit geografickou pozici výskytu plevele. Přesnost (Accuracy) v tomto případě je, když se naměřená data budou shodovat s lokací výskytu plevele. Preciznost (precision) v tomto případě je, když se jednotlivé naměřené hodnoty navzájem od sebe výrazně neliší. Vysoká přesnost hodnot vůbec nic neříká o tom, jak jsou data odchýlena od skutečné pozice plevele. Preciznost vyjadřuje rozptyl jednotlivých měření (Chapman, 2005). S precizností souvisí ještě jeden pojem, rozlišení. Rozlišení vyjadřuje nejmenší základní jednotku, kterou jsme schopni rozlišit v rámci měření. Je to rozlišitelná jednotka našeho způsobu měření a zároveň digitalizace dat do systému. Jak měřící metoda, tak systém musí být schopny požadované rozlišení zpracovat. Na následujících obrázcích jsou znázorněny všechny kombinace vlastností preciznosti a přesnosti.



Obrázek 1: Ukázka rozdílu mezi přesností a precisností. Zdroj: (Chapman, 2005)

Obrázek ukazuje odlišnost přesnosti a preciznosti v geografické podobě. Červený bod je reálná, správná hodnota. Černý bod reprezentuje umístění zaznamenané pozorovatelem.

a) Vysoká preciznost, malá přesnost.
b) Malá preciznost, malá přesnost ukazující náhodnou chybu.
c) Malá preciznost, velká přesnost.
d) Velká preciznost, velká přesnost.

Mimotechnická pomůcka, která také pomáhá k pochopení těchto vlastností, je, že si za přesnost dosadíme pojem „správnost“ a za pojem preciznost pojem „přesnost“. Pak pokud si celý odstavec přečtete s těmito pojmy, bude snáze pochopitelnější.

Na obrázcích je jasné, jakou mají jednotlivé vlastnosti úlohu. Na zadavateli je, aby určil, jaké vlastnosti od pozorovaných a ukládaných dat očekává.

3.1.1.3 Včasnost (Timely)

Tato charakteristika má několik rozměrů a lze se na ni dívat různými pohledy. Základní rozměr je, že daná data je potřeba dodat ve správný čas. Ideálně ne moc brzy a ne moc pozdě. Pokud by data byla dodána s velkým předstihem, mohlo by zde vzniknout riziko hromadění se požadavků ve frontě a pokud by data měla zpoždění, vznikalo by zde riziko, že u následujících procesů by vznikaly prostoje. Dalším významem je, že operace nad danými daty probíhají v očekávaný čas. Tato vlastnost pak dostává další rozměr, a to je úspěšné dokončení daného úkolu. Velmi záleží, jak je tato vlastnost na začátku daného projektu definována.

3.1.1.4 Kompletnost (Complete)

Kompletnost je udaná míra, od které se data považují za dostatečně obsáhlá. A je lepší mít menší počet kompletních dat než nespočet neúplných informací, které nelze použít.

Záznamy musí obsahovat veškeré atributy, aby popisovaný objekt reálného světa byl dostatečně popsán tak, jak je požadováno zadavatelem. (Chapman, 2005)

3.1.1.5 Konzistentnost s jinými zdroji (Consistent)

Konzistentnost se skládá ze dvou částí sémantické a strukturální. Sémantická konzistentnost vyžaduje od dat, aby byla jasná, jednoznačná a důsledná, tj. významově konzistentní. Strukturální konzistentnost vyžaduje, aby data stejných typů atributů byla uložena ve stejné struktuře a v identickém formátu. Tyto konzistentnosti následně zvyšují přehlednost dat, a v důsledku zrychlují práci s daty. Konzistentnost je velmi důležitá vlastnost, která je potřeba zakotvit především v metodických pokynech, kde je stanoveno, jak uživatel má zadávat data do systému. (Redman, 1996)

3.1.1.6 Relevantnost (Relevant)

Relevantnost informace značí její významnost vzhledem k zadaným požadavkům od uživatele. Relevance je také používána při vyhledávání. Zde se porovnává vyhledávaný pojem s nalezenými výsledky a podle míry shody je pak výsledek označen za relevantní a v opačném případě za irelevantní.

3.1.1.7 Obsáhlost (Comprehensive)

Obsáhlost popisuje do jaké míry je dané téma informacemi pokryto tak, aby uspokojovalo uživatele. Obsáhlost můžeme definovat podle počtu atributů popisující reálný objekt nebo podle počtu stránek, které popisují konkrétní situaci, stav atd. Obsáhlost je velmi subjektivní a nelze jednoznačně posoudit, zda informace jsou obsáhlé. (IAIDQ, 2015)

3.1.1.8 Správná úroveň detailu (Proper Level of Detail)

Tato charakteristika je velice subjektivní, protože je v ní stanoveno, na jakou míru detailu má být realita/pozorování zaznamenáno. Požadavek na úroveň detailu může být zadán kvantitativně – kolik atributů, měření atd., nebo kvalitativně – je dán rozsah, který má objekt popsat. (Redman, 2001)

3.1.1.9 Pochopitelnost (Easy to Read)

Určuje, jak vhodně jsou data vyjádřena, zda jsou vhodně dokumentována, zda jsou vyjádřena ve srozumitelných jazycích. Dále také, zda jsou jednotlivé pojmy jednoznačné a výstižné. Pochopitelnost je jednou z charakteristik, která je také do značné míry

subjektivní a velmi záleží na citu a pohledu, podle kterého je hodnocena jako dostatečně splněná. (Chapman, 2005)

3.1.1.10 Jednoduchost interpretace (Easy to Interpret)

Tato charakteristika je velice komplexní a určitě je možné najít dopad této charakteristiky i do mnoha ostatních. Jednoduchost interpretace v sobě obsahuje snadné porozumění, vysvětlení a předání informace tak, aby nebyla zkreslena a bylo předáno to, co je požadováno. Pokud je tato charakteristika opomenuta, hrozí riziko nejednoznačnosti, zmatenosti, neúplnosti, rozporu a chyb, které nelze jednoduše interpretovat (Dalcin, 2005).

3.1.1.11 Platnost (Currency)

Platnost lze přirovnat k „Spotřebujte do“ na potravinových výrobcích. Je to datum, do kdy jsou data použitelná, nebo můžeme použít stravitelná, pokud chceme použít terminologii z potravinářství.

Tato vlastnost koresponduje s poslední změnou neboli aktualizací uživatele. Protože pokud jsou data pravidelně aktualizována, pak jejich aktuálnost „přetrvává“. Data nemusí být vždy aktualizována, ale musí být kontrolována a označena za aktuální, protože ne všechna data stárnou (Chapman, 2005).

3.1.1.12 Zabezpečení (Security)

Zabezpečení dat má úlohu zajistit data před poškozením nebo ztrátou. Zároveň se stará o to, aby přístup k datům byl autorizovaný i autentifikovaný. Jednotliví uživatelé by měli mít pouze odpovídající oprávnění. Zabezpečení dále zajišťuje soukromí a pomáhá při ochraně osobních údajů. (Roebuck, 2011)

3.1.1.13 Obnovitelnost (Recoverability)

Tato vlastnost popisuje schopnost, jak rychle je možno obnovit data ze sekundárního média, pokud není primární medium dostupné z důvodu zničení, selhání, poškození nebo nedostupnosti. Rychlost a hlavně způsob obnovení dat jsou klíčové body, které jsou v této charakteristice definované. (Roebuck, 2011)

3.1.1.14 Ovladatelnost (Manageability)

Tato charakteristika popisuje, jak jsou data vhodně zpracována z funkčního hlediska. Zda je prostředí intuitivní a jestli jednotlivá vstupní pole jsou na těch místech, kde je uživatel očekává. Ovladatelnost pomáhá uživateli orientovat se v novém prostředí. Jsou to

informativní nápovědy, které radí, co má uživatel dělat. Jedná se i o grafickou stránku systému, která uživatele jednoduše vede tak, aby zaměřil svou pozornost na to, co je důležité, a byl schopen rychle s minimálním počtem kliknutí plnit požadované úkoly. (Vaníček, 2010)

3.1.1.15 Účinnost (Efficiency)

Tato charakteristika informace je mírou schopnosti prostředí daného systému, jak efektivně je možno systém spravovat v poměru k odpovídajícím nákladům, k rozsáhlosti obsahu, častosti změn, kvality záznamů a vzhledem k použité infrastruktuře. Jestliže je systém jednoduše spravovatelný, pak lze snadno přidávat, mazat nebo měnit informace bez dopadu na ostatní funkce. (IAIDQ, 2015)

3.1.1.16 Měnitelnost (Changeability)

Měnitelnost je schopnost dat být upravena vzhledem ke svému typu, hodnotě, technologii uložení nebo požadavku. Je to vlastnost, která umožňuje změnu údajů v datech. Udává jaké úsilí je potřeba vykonat, aby byla provedena změna do systému. Změna například může být jednoduše zavedena přes front-end nebo je potřeba zadat požadavek na aplikační podporu systému a další. (Vaníček, 2004)

3.1.1.17 Přenositelnost (Portability)

Toto je vlastnost, která dovoluje softwaru být používán/instalován na různých prostředích bez nutnosti úpravy. Předpokladem je univerzálnost jednotlivých používaných rozhraní, které abstrahují od samotných funkčností. Pokud jsou funkčnosti jednoduše přenositelné napříč systémy, výrazně to snižuje náklady na jejich údržbu. (Roebuck, 2011)

3.1.1.18 Produktivnost (Productivity)

Produktivnost určuje, jak velké množství dat je možno zpracovat za danou jednotku času. Produktivnost může být dále vztažena k použitému hardware, software atd. Nelze také srovnávat produktivnost napříč různými prostředími. Produktivnost závisí na mnoha faktorech týkajících se konkrétních informačních systémů. (Roebuck, 2011)

3.1.1.19 Bezpečnost (Safety)

Tato charakteristika do značné míry souvisí se zabezpečením. Zabezpečení má na starosti autorizované přístupy a bezpečnost a zajišťuje, aby nikdo neoprávněný neměl přístup k datům. Jedná se jak o bezpečnost softwarovou, hardwarovou, tak o bezpečnost objektu.

Charakteristika zabezpečení se do značné míry prolíná s bezpečností a někdy může být zaměňována. (Vaníček, 2010)

3.1.1.20 Důvěryhodnost (Credibility)

Důvěryhodnost je pohled na kvalitu dat vzhledem k jejich zdrojům. Důvěryhodnost zdrojů je posuzována na základě historie samotné informace nebo dobré pověsti zdroje. Ověřování zdrojů probíhá různými způsoby. (IAIDQ, 2015)

3.1.1.21 Dostupnost (Availability)

Míra udává dohledatelnost dat v systému. Tato charakteristika souvisí s flagy (značkami), které označují jednotlivá data a zjednodušují dohledatelnost dat podle klíčových slov. Tato charakteristika se do velké míry týká metadat, která popisují samotná data. V podstatě se jedná o rejstříky, které zjednodušují vyhledávání. (Vaníček, 2010)

3.1.1.22 Shoda s předpisy (Regulatory Compliance)

Shoda s předpisy udává, jak systém splňuje požadavky na normy, na legislativní požadavky a na právní požadavky.

Popisy charakteristik kvality dat nastínily, co všechno je sledováno v rámci atributů kvality dat. Existují další oblasti, které také ovlivňují kvalitu dat, ale jsou spíše technického charakteru a nelze je přiřadit k jediné charakteristice. Jedná se o oblasti technické, oblasti technologického zpracování dat a o zvolené principy ukládání dat do úložišť.

Jednotlivá rozhodnutí týkající se této oblasti závisí na zvoleném datovém modelu a principu ukládání dat, jedná se tedy o architektonická rozhodnutí ohledně informačního systému a dopady na zvolené technologie a využití jejich možností. Jak bylo zmíněno v úvodu, bude se tato práce výlučně zabývat technologiemi perzistentních úložišť a jejich možnostmi se zaměřením na zvyšování kvality dat.

3.1.2 Formáty dat

Kvalita dat je ovlivňována formátem dat, ve kterých jsou informace uložené. Ovlivňována je zde zejména přístupnost, obnovitelnost, účinnost, měnitelnost a další. Tyto charakteristiky souvisí se správou informací. Proto je třeba zvolit na základě požadavků odpovídající formát dat. Základní datové formáty budou popsány v následujících odstavcích.

Aby bylo možné s daty pracovat, musí být tato data z nějakého úložiště nahrána. Uživatele samotné nezajímá, zda se jedná o cloudové úložiště nebo mirorovaný lokální disk. Pokud ale zohledňujeme kvalitu dat, je třeba zohlednit i tuto oblast ovlivňující kvalitu dat.

Nejprve budou popsány způsoby ukládání dat v IS. Velmi zjednodušeně lze říci, že se jedná o datové soubory a databáze. Dále můžeme najít ještě další specifické způsoby ukládání různých typů dat, ale ty jsou používány hlavně ke specifickým účelům, proto zde nebudou dále zmíněny.

3.1.2.1 Datový soubor

Datové soubory můžeme dále dělit na:

- Plain text soubor – jednoduchý txt soubor.
- CSV soubor – txt soubor, zde jsou jednotlivé hodnoty odděleny určitým separátorem, samotným hodnotám může předcházet hlavička, která určuje typy hodnot v jednotlivých sloupcích, navíc hodnoty mohou být ohraničeny uvozovkami.
- YAML soubor – strukturovaný soubor hodně používaný internetovými technologiemi.
- XML soubor – strukturovaný soubor používaný pro přenos souborů, například je používán webovými službami.
- Soubory kancelářských balíků (doc,xls…).
- Obrázkové, audio, video soubory, další.

Pro kvalitu dat je důležité, aby bylo možné tyto jednoduché soubory dat určitým způsobem strukturalizovat - vlastnost strukturovaná konzistentnost. Popřípadě i definovat rozsah a typy jednotlivých proměnných a navíc, by mělo být možné s těmito soubory pracovat jako s objekty. Ty je možné načíst a dále s nimi reálně jako s objekty pracovat – validace při objektové konverzi. Navíc je možné již při převodu ze souboru na objekt použít některé validace, ale záleží na požadované přísnosti kontrol na vstupu. (Chapman, 2005)

Na základě výše uvedených požadavků se z vyjmenovaných typů souborů hodí pouze soubor typu XML nebo s určitými omezeními i YAML. Je ale zavádějící si myslet, že kvůli kvalitě dat by byly tyto formáty upřednostňovány. Další zmíněné formáty

se používají také v hojné míře, ale každý specifickým způsobem. Například formát CSV se používá pro předávání reportů.

Soubor typu XML dovoluje nastavení různých omezení, výčtů typů dat, vnořených struktur atd. Toto je velmi zajímavé ve spojitosti s kvalitou dat. Pokud jsou data strukturována, je možno data rychle validovat a následně upravit na požadovanou strukturu.

YAML formát je hodně specifický, protože je strukturovaný, ale na druhou stranu velmi dobře čitelný i pro člověka, což bylo jeho účelem. A primárně byl tento formát navržen pro serializaci objektových dat. Tento jazyk je hodně používán v rámci programovacího jazyka Ruby a dalších webových technologií.

3.1.2.1.1 Soubor typu XML

XML (Extensible Markup Language) (Herout, 2007) (Buxton, 2006) je textový soubor, který je tvořen hierarchickou strukturou tagů, které se mohou navíc vzájemně zanořovat a také obsahovat různé atributy. Tuto kompletní strukturu lze i kontrolovat přes speciální formát souboru. Buď pomocí DTD (zastaralé) nebo XSD. XSD je rozšířenější hlavně z toho důvodu, že je, stejně jako XML, tvořeno tagy v hierarchické struktuře. Pravidla pro XSD schéma jsou velmi komplexní a umožňují validovat podle rozličných pravidel.

XML formát se může zdát být neohrabaný, ale hierarchická struktura a možnost její kontroly a řízení z něj dělá velmi komplexní a mocný nástroj. Navíc je možné XML soubor přemapovat na objekt a následně s takovým objektem dále pracovat. V Javě se pro toto mapování například používá technologie JAXB.

Další vlastností XML je možnost přímého uložení do databáze. A to buď do relační, nebo XML databáze.

Tento souborový formát je navrhnut pro komunikaci na síti, respektive pro volání webových služeb, ale pro perzistentní uložení klientských dat se většinou nehodí. Ukládání dat v souborových systémech je málo strukturované a nelze zde provádět reporty nebo kontrolovat určité vlastnosti atributů jednotlivých souborů, proto je přístup k jednotlivým informacím více těžkopádný. Určitým řešením je kombinace těchto formátů s databází, kde atributy jsou již indexované a vyhledávání v databázi je již použitelnější.

Další výhodou strukturovatelnosti souboru je dobrá čitelnost jak strojem, tak uživatelem. A strukturované soubory jsou jednoduše serializovatelné a následně je lze jednoduše poslat

do jiného systému. Proto při interakci mezi systémy je strukturovanost informace velmi důležitá.

3.1.2.2 Databáze

Druhou kategorií perzistentních úložišť jsou databáze. Jedná se o aplikaci, která řídí přístup k datům až na úroveň jednotlivých atributů, umožňuje informace strukturalizovat a vkládat do databáze požadavky na validace, omezení a konzistentnost tak, aby data splňovala všechny požadavky na kvalitu. Dříve, než budou popsány konkrétní vlastnosti databází, bude uvedeno rozdělení podle typů databází:

- Relační databáze
- Objektová databáze
- XML – databáze
- Dokumentové databáze

Dále pak existují specifické typy databází, které jsou obsaženy ve specifických programech a většinou jsou laděny pro specifické účely, například různé znalostní systémy. Existují také databáze, které jsou vyvinuty pro jeden konkrétní účel, ale v konečném důsledku je relativně sporné, zda se vůbec ještě jedná o databázi nebo velmi specifickou službu. Takovým příkladem je služba github a další. Ukázka vyjmenovaných nástrojů naznačuje, že by v praxi bylo možno najít velké množství úložišť typu databáze, ale v mé práci se pouze omezím na základní vyjmenované databáze, které budou dále popsány.

3.1.2.2.1 Relační databáze

Pod pojmem databáze si člověk představí nejčastěji relační databázi (Elliott, 2015), (Harrington, 2010), (Lacko, 2013). Tato databáze se skládá z dvourozměrných tabulek (relací) a tyto tabulky lze za běhu vzájemně pomocí relací propojovat a vytvářet tak vazby mezi daty. Jednotlivé tabulky jsou pak tvořeny sloupci, které mají definovaný typ, velikost a řádky, které obsahují v jednotlivých sloupcích data, čímž dostáváme strukturované informace. Z pohledu na datovou strukturu jsou takto dělené informace jednoduše pochopitelné a přehledné, což je i hlavní výhoda tohoto typu databáze.

Tento typ databází je využíván ve firmách, kde je spravováno velké množství dat. A díky tomu, že relační databáze prošly dlouhým vývojem, nabízejí dnes celou škálu rozšíření

a dalších služeb. Navíc je možné přímo na úrovni databáze programovat samotnou logiku, která pak pracuje přímo s daty v tabulkách a není zde žádné další rozhraní na jiný systém.

3.1.2.2.2 Objektové databáze

Objektové databáze představují alternativu k relačním databázím (Harrington, 2000), (Bancilhon, 1992). Tento typ databází nabízí programátorovi jednodušší správu a práci s uloženými daty. Není potřeba řešit mapování nebo jiný typ serializace pro možnost následného uložení objektů. Programátor jednoduše uloží objekt, se kterým pracuje. Navíc je možné na objekt vázat funkce/procedury, které se s objektem ukládají. Vazby zde také není potřeba extra řešit, jedná se buď o součást objektu, nebo odkaz na existující objekt. Podobně jako sloupce v databázi nebo atributy v XML je možné definovat atributy na objektu včetně jejich velikosti a typu. Jedná se pouze o jiný pohled na data, také velmi dobře definovatelný a strukturovatelný, proto i zde je při přesném definování požadavků na data splněna kvalita dat. Samozřejmě se jedná pouze o kvalitu struktury dat, nejedná se o jejich smysluplnost a hodnotu.

K objektovým databázím nabízí relační databáze určitou alternativu, kterou je objektově – relační mapování, kdy se zavádí konverze mapování objektů do řádků tabulky, a tak se docílí jednodušší práce s objekty.

3.1.2.2.3 XML databáze

Soubor typu XML byl již zmíněn výše. Jedná o ukládání XML souborů do databáze, kde základní jednotkou je právě tento formát (Powell, 2007). Protože tento formát je sám o sobě již strukturovaný a obsahuje metadata – tagy, které popisují samotné informace, už tato samotná struktura vybízí k uložení a oindexování a není potřeba vytvářet speciální datový model. XML databáze můžeme rozdělit jednoduše podle toho, zda formát XML je nativně databází podporován nebo se jedná například pouze o relační model s uloženými a oindexovanými daty typu XML.

Tento model má nespornou výhodu v tom, že XML formát je nativně používán v mnoha službách např. webové služby. Proto přímé uložení tohoto formátu bez konverze zrychluje práci s těmito daty. Navíc pokud je struktura XML podrobně definována, obsahuje XML strukturované a typově definované proměnné, které jsou co do členění a strukturování velice kvalitní a navíc i pro člověka velmi jednoduše čitelné. Tento typ databáze je vhodný

pro ukládání dat, které jsou již primárně ve formátu XML, například již zmiňované webové služby.

3.1.2.2.4 Dokumentové databáze

Jsou jedním z hlavních představitelů NoSQL databází, které využívají pro své ukládání dokumenty podobné formátu JSON (Chodorow, 2013). Dokumentové databáze se zaměřují na rychlost a hlavní výhodou je dynamické vytváření databázového schématu, což je velmi užitečné v případě ukládání internetových stránek, kde je tento typ databází často používán. U tohoto typu databází se ukládaná data nemusí fixně vázat na předem danou strukturu (tabulku), ale ukládá data do dokumentů, což přispívá k intuitivnějšímu použití. Jedním z velkých omezení je nesplnění požadavků na ACID při změně dat. Proto se tyto databáze nevyužívají u firemních systémů, kde toto omezení vadí. Jsou ale používány v backendu velkých webových serverů, kde je požadována hlavně rychlost zpracování a nevadí zmiňovaná omezení. (mongoDB, 2016)

3.1.2.3 Výhody databázových systémů před jednoduchými soubory dat

Jednou z hlavních výhod databázových systémů před jednoduchým souborem dat je jejich pokročilá správa a díky tomu snadnější možnost udržování kvality dat. Porovnávat tyto dvě řešení, není prakticky možné. V konečném důsledku se i samotná databáze skládá ze souborů, kam ukládá data. To se ale již týká architektury dané databázové aplikace a jejího konkrétního technického řešení. Hlavním rozdílem těchto dvou nesourodých řešení je to, že v rámci souborů jsou data z pohledu systému nestrukturovaná, ale v databázi se skládají z jednotlivých atributů a jejich vzájemných provázaností.

Mohlo by být namítnuto, že načtený soubor XML v paměti je již strukturovaný, což je pravda, ale háček je v potřebě daného načtení. Tuto překážku řeší již zmiňované XML databáze. Dalších typů databází, které se skládají z určitého typu souborů, bychom našli ještě více, ale pro uvedení do tématu databází je výčet dostačující.

Dále budou vyjmenovány vlastnosti a možnosti databází, které podporují kvalitu dat a v konečném důsledku tak zvyšují použitelnost samotných informací pro business. U vlastností zde zmíněných vycházím ze zkušeností ve spojitosti s praktickou částí a s relačními databázemi, ale ostatní typy databází také postupně získávají zmiňované funkčnosti, ať ve stejné nebo v omezené míře. Aktuálně je stále nejrozšířenější relační databáze, proto i praktická část bude v tomto typu databází realizována. (solid IT, 2016)

3.1.3 Kvalita dat – součástí firemní kultury

Kvalita dat nutí organizaci přemýšlet o datech v dlouhodobém horizontu, motivuje odpovědné osoby rozhodovat tak, aby se kvalita dat zvyšovala. Dále formalizuje přístup k datům, aby byla zajišťována míra informace daných dat, která následně maximalizují podporu podnikání společnosti. Dále přispívá ke zlepšení přístupů k datům, aby byla tak snižována rizika zanášení chybných nebo nevalidních dat do systémů. (Chapman, 2005)

Data nemohou být vytrhávána z kontextu dané firmy a businessu, který daná firma dělá. Proto i pohled na kvalitu dat je potřeba zohlednit v celé firemní kultuře i firemní politice. Pohled na kvalitu dat napříč celou firmou je novou perspektivou v kvalitě dat, protože do teď byla zmiňována kvalita dat především přes pohled na uživatele. Pokud by byl pohled pouze omezen na uživatele a jeho úlohu v rámci kvality dat, nebylo by to dostatečné. Proto je třeba kvalitu dat zakotvit už ve firemní politice, aby ovlivňovala management, modelování, analýzu, řízení, ukládání dat a další. (Chrisman, 1991)

Aby kvalita dat, respektive i čištění dat odpovídalo potřebám a požadavkům podnikatelské činnosti dané firmy, je potřeba, aby v rámci ní fungovala spolupráce všech zainteresovaných stran. Zadavatelé musí být schopni komunikovat a předávat informaci o tom, jaká je představa ohledně kvality dat a co od těchto dat business firmy očekává, tj. co je v datech uloženo, s jakou přesností se budou hodnoty proměnných ukládat, v jaké míře a granularitě je daná informace žádána, jaký formát by měl být pro uložení zvolen, jaký typ databáze je pro uložení dat nejlepší a jaké technologie jsou pro IS použity. Dalšími požadavky může být to, s jakými dalšími zdroji informací má daný IS firmy spolupracovat, jaká data bude dostávat ať od uživatelů nebo klientů dané společnosti. Proto, aby se docílilo jednak maximálního uspokojení uživatelů a zadavatelů systému, je potřeba zajistit kvalitní spolupráci zainteresovaných stran při specifikaci kvality a obsahu dat a současně tyto požadavky jednoznačně vysvětlit příslušným pracovníkům IT. Tyto zmiňované požadavky by měly být definovány na úrovni IT architektury.

Podle Chapmena (2005) je v rámci firemní politiky potřeba vylepšit, zefektivnit a zrychlit komunikaci jak s uživatelem, který data zpracovává, tak s klientem (poskytovatelem) zdrojových dat, aby došlo k harmonii a maximálnímu využití potenciálu všech informací. Dále v rámci politiky musí být prosazován náhled na data v širším kontextu, aby mohla být využita v dlouhodobějším horizontu. Zajišťování této oblasti ve firmě má na starosti Quality management.

3.1.4 Management kvality (Quality Management)

Quality management je management rozhodující o kvalitě. Jako každý jiný management je zodpovědný za rozhodnutí, zda kvalita dat bude odpovídat daným požadavkům. V nejjednodušším pohledu se jedná o to, aby při akceptacích nevznikala vyšší chybovost, než je požadováno. Kontrolní mechanismy kvality můžeme zjednodušeně rozdělit na vnější a vnitřní, kde vnitřní kvalitu má na starosti quality Control a vnější kvalitu má na starosti quality Assurance. Obě dvě oblasti spolu těsně souvisí a někde se i překrývají. (Chapman, 2005)

3.1.4.1 Řízení kvality (Quality Control)

Tato oblast je součástí quality managementu, který má na starosti naplňování požadavků na kvalitu v rámci firmy, což je řečeno i v definici podle ISO 9000. (Vaníček, 2010)

Quality control se nesoustředí pouze na produkt, který nás bude především zajímat, ale na všechny možné požadavky napříč organizací. Řízení kvality se zabývá procesy, popisem jednotlivých náplní zaměstnanců, výkonem a návazností jednotlivých článků v procesu výroby produktu. Dále se zabývá znalostmi zaměstnanců, jejich dovednostmi a v návaznosti na jejich vzdělávání i rekvalifikaci. Řízení kvality se také zabývá personálními záležitostmi, firemní kulturou a dalšími procesy, které nesouvisí se samotnou výrobou produktů firmy. (Soley, 2016)

Nás zajímá pouze stanovování kvality dat a potažmo kvality informačních systémů, kde je kontrolována chybovost, nepřesnost vůči specifikovaným požadavkům zaznamenaných ve smluvních závazcích. Kontroly a ověřování kvality se týkají zejména při dodávání nových funkčností do systémů nebo předávce nových informačních systémů.

3.1.4.2 Zabezpečování kvality (Quality Assurance)

Jedná se o kontrolu dodržování procesů kvality ve firmě. Je to jakýsi vnitřní audit, který prověřuje a reportuje managementu stavy týkající se kvality. Tato oblast je také součástí quality managementu a velmi úzce souvisí s quality control, jen to není řídící část, ale kontrolní část. Jak i překlad napovídá, jedná se o „zabezpečování kvality“. Quality assurance se stará o to, aby při předávání produktu klientovi, byly splněny dohodnuté požadavky na produkt. Hlídá dodržování požadavků na kvalitu, tj. produkt nesmí obsahovat žádné chyby, které by bránily jeho používání nebo by snižovaly jeho použitelnost. (Soley, 2016)

3.1.4.3 Metodické přístupy a pokyny přispívající ke kvalitě dat

Náplní quality managementu je zajišťování kvality dat, jak bylo zmiňováno výše. Aby kvalita dat byla udržována, je potřeba u dat stanovit kompetence a odpovědnosti. Podle Redmena (2001) je nejlepší volbou přiřadit zodpovědnost za kvalitu dat tomu, kdo data vytváří/zakládá. A pokud toto není možné, tak má být přiřazena odpovědnost co nejblíže tomuto uživateli. (Redman, 2001)

Jinak řečeno, člověk přicházející do kontaktu s daty má o nich i největší přehled a odpovědnost je potřeba vyžadovat proto, aby vždy bylo možné určitého člověka urgovat a případně žádat o sjednání nápravy.

Dalším důležitým přístupem v oblasti udržování kvality dat je prioritizace požadavků. Toto je důležitý přístup proto, abychom byli schopni dosáhnout maximální hodnoty dat pro nejvíce uživatelů v nejkratším možném čase. Zejména je potřeba prioritizovat vkládání a validaci dat, aby se minimalizovalo množství chybných nebo nevalidních dat ukládaných do systému. Místo, kde jsou chybná data vkládána do systému, musí být co nejdříve identifikováno a opraveno a zároveň je potřeba udělat vše proto, aby se toto již neopakovalo.

A aby prioritizace měla maximální přidanou hodnotu, je třeba ji provádět systematicky. Ostatně systematičnost musí být zakotvena již na úrovni vize podniku a nejen v quality managementu, aby veškerá rozhodnutí směřovala k jednomu cíli. K cíli danému, jednoznačnému a všeobecně v podniku známému. Špatná prioritizace a nesystematické rozhodování má za následek zmatek a neefektivně vynaložené prostředky. (Chapman, 2005)

3.1.4.4 Prevence je lepší než reaktivní přístup opravy chyb

Dalším dobrým tipem je proaktivní přístup k opravě chyb. Náklady na následnou opravu chyb v databázích mohou být značné. Známé a zřejmé chyby v datech musí být opraveny, jakmile je to možné a musí být hned jasné, kdo tyto chyby má opravit, viz výše zmíněné kompetence a odpovědnosti. Průběžné kontroly stojí pouze zlomek nákladů oproti nákladům, které vznikají na základě následků. (Redman, 2001) Navíc při ponechání chybných dat v systému mohou být tato data chybně používána a způsobovat tak sekundární následky a zkreslení. Na základě chybných analýz mohou být učiněna špatná rozhodnutí, která mají dalekosáhlé následky. (Chapman, 2005)

Proto opravě chyb v core systémech musí být přiřazena největší priorita, aby nedocházelo k přenosu chyb do dalších přidružených systémů a nerostly tak náklady na odstranění většího množství chyb, než které byly na počátku.

3.1.4.5 Kooperace a spolupráce – systematický přístup

Stejně jako v jakékoliv jiné činnosti je zbytečné dělat jednu věc na více místech. Stejně tak i práce s totožnými daty by měla být co nejvíce zkonsolidována a větší úsilí by místo toho mělo být koncentrováno na kvalitu zpracovávání dat. S kooperací zároveň souvisí i spolupráce. Tato spolupráce by neměla začínat u samotných pracovníků, ale měla by být podporována už samotným vedením. Pokud budeme přistupovat i k samotnému řízení organizace systematicky, bude se to odrážet i na vyšší kvalitě dat, díky jasným kompetencím a celkovému pořádku ve firmě.

V rámci systematického přístupu by měly být zakotveny pokyny pro uživatele aplikace, aby poskytovali rychlou zpětnou vazbu o datech a přispívali tak ke zvyšování kvality dat. (Chapman, 2005)

3.1.5 Management celkové kvality dat (TDQM – The Total Data Quality Management)

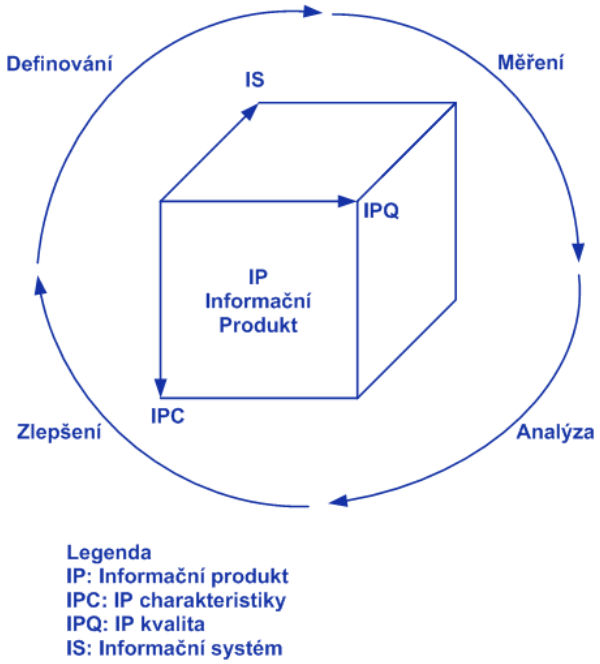
TDQM obsahuje jednotlivé výše zmiňované koncepty a přístupy, které jsou obecně zapracovány do kompletního managementu zohledňujícího kvalitu dat napříč celou firmou (Kostiha, 2012). Je to standardizované zapracování jednotlivých best practise do jedné ucelené metodiky. Hlavním principem této metodiky je dodat zákazníkovi produkt o vysoké kvalitě. Této vysoké kvality je dosahováno díky kontinuálním cyklům postupného vylepšování založeného na zpětné vazbě. (Wang, 2002)

TDQM bere v potaz výše zmiňované, že kvalita dat, pokud je brána vážně, musí být zakotvena v podnikových vizích. Navazující manažerské strategie a rozhodnutí proto musí kvalitu dat brát také vážně.

Tato metodika je založena na zpětné vazbě. Díky zpětné vazbě je možné iterativně zlepšovat kvalitu dat. Tuto zpětnou vazbu dostáváme v průběhu cyklu, který se skládá ze čtyř částí:

- Definování (Define)
- Měření (Measure)

- Analýza (Analyze)
- Zlepšení (Improve)



Obrázek 2: Cyklus TDQM. Zdroj: (Wang, 2002)

Před popsáním jednotlivých částí cyklu, budou definovány role, které tato metodika stanovuje:

- Information supplier (dodavatelé informací) – uživatelé, kteří vkládají data do systému.
- Information manufacturer (správce systému) – programátoři a administrátoři, kteří se starají o vzhled, vývoj nebo podporu systému a systémovou infrastrukturu informačního produktu.
- Information consumer (uživatelé informací) – uživatelé, kteří s informačním produktem pracují.
- IP manager (manažer informačního produktu) – manažeři, kteří jsou odpovědni za řízení IP výrobního procesu napříč IP životními cykly.

Při definování rolí zazněl výraz „Informační produkt“. V rámci terminologie TDQM se jedná o produkt, který je tvořen v rámci výrobního procesu informačním výrobním systémem. Koncept informačního produktu je používán, aby byl zdůrazněn fakt, že výstupní informace z výrobního informačního systému mají hodnotu, která je přenositelná k zákazníkovi. Dále bude používána zkratka IP. (Wang, 2002)

3.1.5.1 Fáze definování (define)

Tato fáze definuje první část cyklu. Jedná se o definici IP charakteristik, definici požadavků na kvalitu informací a definice informačního systému. Pod **definicí IP charakteristik** se skrývá definování entit. Toto definování probíhá ve dvou úrovních. Nejprve na high-level úrovni, kdy jsou vyjmenovány entity, které uživatel (information consumer) potřebuje pro uchovávání informací. Na příkladu bankovního účtu se jedná o účet klienta, ke kterému je účet veden apod. Na nižší úrovni jsou pak definovány podrobnější vlastnosti jednotlivých pojmenovaných entit. Zároveň jsou také určeny vazby mezi těmito entitami. Pokud bychom tyto vztahy a vlastnosti chtěli znázornit graficky, byl by pro to použit E-R diagram.

Dalším krokem v této fázi je **definice požadavků.** Definice požadavků se tvoří na základě pohledů a priorit od dodavatele systému, uživatele informací a správce systému. Zástupci jednotlivých rolí dávají požadavky dle kategorií a jednotlivých dimenzí. Tyto dimenze jsou synonymem výše zmíněných atributů nebo také vlastností. V rámci TDQM jsou stanovené tyto kategorie a jejich dimenze:

- Vnitřní IQ (Intrinsic)
    - Přesnost
    - Objektivita
    - Věrohodnost
    - Reputace
- Přístupnost IQ (Accessibility)
    - Přístupnost
    - Zabezpečení
- Kontextová IQ (Contextual)
    - Relevance
    - Přidaná hodnota
    - Včasnost
    - Kompletnost
    - Množství dat
- Reprezentační IQ (Representational)
    - Interpretovatelnost
    - Srozumitelnost

- Stručnost
- Reprezentace
- Konzistentní reprezentace

Je patrno, že se jedná o určitou obměnu v pohledu na jednotlivé vlastnosti, které byly definovány výše. Proto na tomto místě nebudou více rozepisovány.

Posledním krokem ve fázi definování je krok **Definice výrobního informačního systému**. V tomto kroku jsou definovány vztahy mezi rolemi a výše definovanými entitami. V rámci těchto vztahů se následně definují odpovědnosti a procesy, které budou rozhodující pro vytvoření informačního produktu. (Wang, 2002)

3.1.5.2 Fáze měření (Measure)

Tato fáze slouží k definování metrik pro ohodnocení kvality informací. Jedná se o stanovení postupů a metod, jak budou jednotlivé požadavky stanoveny a dodržovány jejich dimenze. Závisí na konkrétních přiřazených rolích a prioritách, jak budou metriky stanoveny a zároveň i hlídány. (Wang, 2002)

3.1.5.3 Fáze analýza (Analyze)

Výsledky z fáze měření jsou analyzovány a vyhodnocovány pro zjištění aktuálních problémů v kvalitě informací. Metody pro analýzu jsou různé od jednoduchých po komplexní. Některé se zaměřují na správnost metrik a jejich efektivnost vzhledem k měřené dimenzi, jiné analyzují využívání zdrojů jako takových a výhodnost využití těchto zdrojů vzhledem k výstupům. Tyto výstupy jsou opět reflektovány pomocí metrik. Fáze analýzy vysvětluje příčiny vzniku nedostatků ve výsledné úrovni kvality a dává podněty k možným vylepšením v dalším cyklu. (Wang, 2002)

3.1.5.4 Zlepšení (Improve)

Fáze zlepšení začíná po dokončení všech analýz předchozí fáze. V této fázi je potřeba identifikovat klíčové oblasti, které je možné vylepšit. Jedná se například o tok informace nebo proces související s informačním systémem nebo o nové uspořádání charakteristiky související s potřebami businessu. (Wang, 2002)

3.1.6 Standardy

Dalším pohledem na kvalitu jsou standardy kvality dat. Jedná se o další přístup ke kvalitě, který zajišťuje dalším způsobem kvalitu dat. Standardy jsou velmi důležité a užitečné,

protože shrnují jednotlivé přístupy a pohledy a vytvářejí co možná nejuniverzálnější metodiky pro zabezpečení kvality dat.

Takovým významným standardem v oblasti softwarové kvality dat je projekt SQuaRE – Software Quality Requirements and Evaluation, který je definován v normách ISO/IEC 25000 až ISO/ 25099. (Vaníček, 2010)

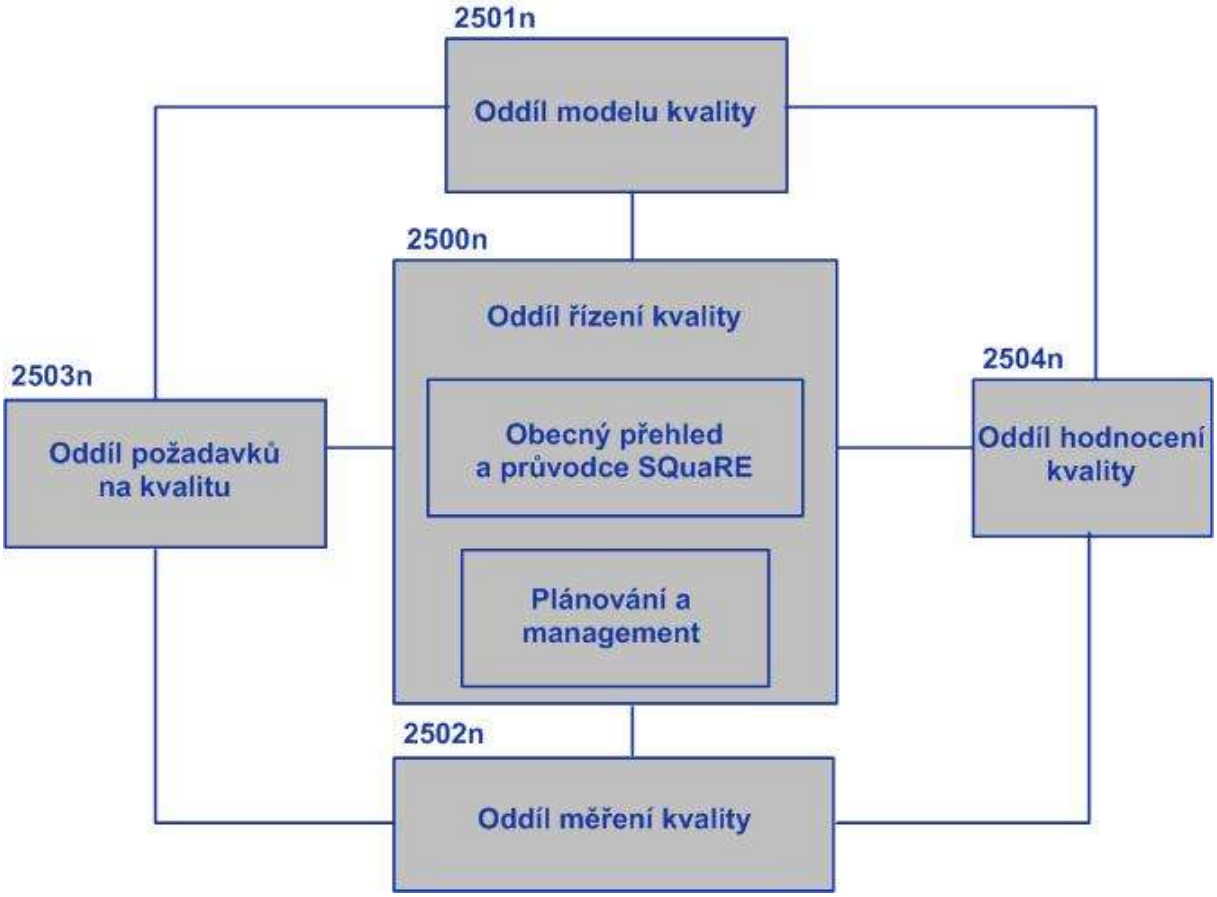
3.1.7 Norma SQuaRE – ISO/IEC 250xy

Norma SQuaRE se skládá z těchto částí:

- Obecná část - 2500n (Vaníček, 2004)
    - Obecný přehled a průvodce po SQuaRE (25000) – zastřešující dokument, definující pojmy a terminologii.
    - Plánování a management (25001) – obsahuje informace pro plánování a řízení projektů hodnocení jakosti.
- Model jakosti - 2501n (Vaníček, 2004)
    - Model jakosti (25010) – popisující model jakosti, definující charakteristiky podcharakteristiky a definující převod uživatelských požadavků na požadavky na produkt.
- Míry pro jakost - 2502n (Vaníček, 2004)
    - Referenční model a průvodce mírami (25020) – definuje informace o mírách jednotlivých atributů,
    - Primitiva pro měření (25021) – základní míry, které lze na produktu měřit přímo (pouhým pozorováním).
    - Vnitřní míry (25022) – kvalifikovaný výběr měr z aktuální technické zprávy 9126-3.
    - Vnější míry (25023) – kvalifikovaný výběr měr z aktuální technické zprávy 9126-2.
    - Míry pro jakost při užití (25024) – kvalifikovaný výběr měr z aktuální technické zprávy 9126-4.
    - Dokumentace hodnotících postupů (25025)
- Požadavky na jakost - 2503n (Vaníček, 2004)

- Požadavky na jakost (25030) – obecné údaje o typech požadavků a zásad, které upřesňují požadavky na tak zvanou „vnitřní jakost“, „vnější jakost“ a „jakost při použití“.

- Hodnocení jakosti - 2504n (Vaníček, 2004)
  - Přehled o procesech hodnocení (25040) – zásady hodnocení produktů z různých pohledů.



Obrázek 3: Skladba norem řady ISO/IEC 250xx - projekt SQuaRE. Zdroj: (Vaníček, 2010)

Podle normy SQuaRE je kvalitní software takový, který má schopnost uspokojovat stanovené a předpokládané potřeby při jeho použití za stanovených podmínek. Na základě zkušeností bylo stanoveno pro model kvality produktu osm charakteristik: funkčnost, bezpečnost, interoperabilita, bezporuchovost/spolehlivost, použitelnost, účinnost, udržovatelnost a přenositelnost. Tyto charakteristiky se používají pro externí a interní kvalitu. Pro kvalitu použití je definováno pět charakteristik: efektivnost, efektivita, uspokojení, bezpečnost a použitelnost. (Vaníček, 2010)

3.1.7.1 Model kvality

Model kvality dělí softwarový produkt do jednotlivých kategorií – charakteristik, které se dále dělí na subcharakteristiky a ty se rozpadají na atributy kvality.

Model kvality definuje dva rozdílné pohledy na kvalitu:

- Kvalita použití
- Vnitřní kvalita a vnější kvalita

Kvalita použití měří kvalitu systému při standardním použití specifickými uživateli pro specifické úkoly. Tato kvalita užití určuje, jak systém umožňuje uživateli provést jeho specifické úkoly.

Vnější kvalita je pohledem na systém jako na černou skříňku, kde je zkoumáno, zda specifické úkoly jsou vykonávány podle zadání, aniž by bylo zřejmé, jaké procesy se v systému provádí.

Vnitřní kvalita je pohledem na systém jako na bílou skříňku. Tato kvalita je většinou zkoumána během vývoje a jsou zkoumány specifické konkrétní funkčnosti a splnění požadavků očekávaných od zadaných funkčností.

Model kvality můžeme dále rozlišovat na určité části softwaru. Jedná se o tyto kategorie:

a) Model vnější a vnitřní kvality softwarového produktu
b) Model kvality systému pro použití
c) Model kvality dat

V každé kategorii jsou opět definované charakteristiky i jejich rozpad, ale tyto charakteristiky jsou již specifické pro danou část softwaru.

V další kapitole se zaměříme pouze na rozpad charakteristik pro model kvality dat, protože klientské záznamy jsou podmnožinou dat. A navíc velká část charakteristik je shodná pro všechny kategorie.

3.1.7.2 Model kvality dat

Pod pojmem software si většinou představíme program, který zpracovává data. Ale v dnešním světě mnoho informačních systémů data neukládá, ale sdílí a data jsou používána různými systémy. Proto v konečném důsledku již nejsou data vázána

na konkrétní systém a je tedy smysluplné vytvořit pro data oddělený model kvality. Tento model kvality je zpracován v mezinárodním standardu ISO/IEC 25012. V rámci modelu kvality dat jsou definovány dva pohledy na tuto oblast. Prvním je vlastní kvalita dat a druhým pohledem je rozšířená kvalita dat.

Vlastní kvalita dat je souhrn podstatných vlastností dat, které souvisejí se schopností uspokojovat stanovené a předpokládané potřeby, pokud jsou údaje používány za stanovených podmínek nezávisle na ostatních systémech. Vlastní kvalita dat se zaměřuje na data samotná.

Rozšířená kvalita dat je definována jako souhrn charakteristik údajů, které souvisejí s jeho schopností uspokojovat stanovené a předpokládané potřeby. Tyto potřeby jsou definovány a následně využívány ostatními systémy ve specifickém kontextu užití. Rozšířená kvalita dat se například zaměřuje na to, aby data byla použitelná, přístupná, jednoznačná… Toto je důležité proto, aby bylo možné data využívat napříč různými ostatními systémy.

Charakteristiky	Kvalita dat	
	Vlastní	Rozšířená
Přesnost	X	
Úplnost	X	
Konzistentnost	X	
Důvěryhodnost	X	
Aktuálnost	X	
Přístupnost	X	X
Shoda	X	X
Důvěrnost	X	X
Výkonnost	X	X
Preciznost	X	X
Sledovatelnost	X	X
Srozumitelnost	X	X
Dostupnost		X
Přenositelnost		X
Obnovitelnost		X

Tabulka 1: Charakteristiky modelu kvality dat. Zdroj: (Vaníček, 2010)

Tabulka obsahuje patnáct charakteristik modelu kvality dat a je z ní patrné i přiřazení jednotlivých charakteristik k jednotlivým pohledům na kvalitu dat.

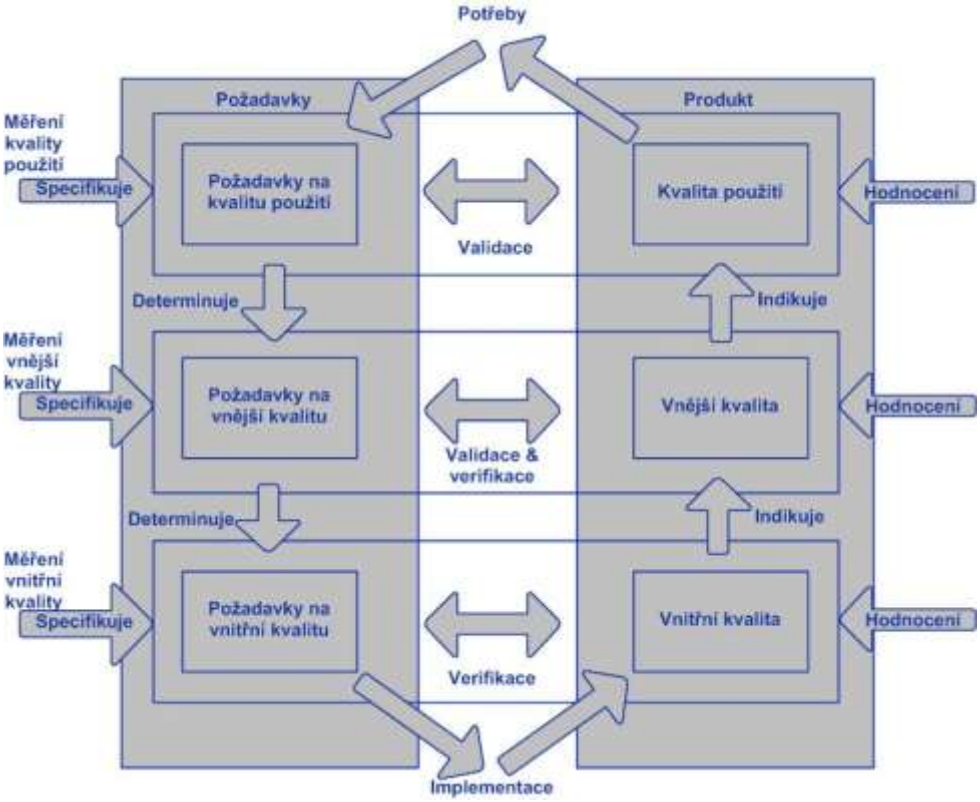
Jednotlivé výskyty symbolů X ve sloupcích značí, zda daná charakteristika je definována pro kvalitu dat vlastní/rozšířenou a je tedy zohledněna ve výsledných měřeních, které určují kvalitu dat. (Vaníček, 2010)

3.1.7.3 Model kvality měření

Model kvality měření navazuje a často odkazuje na model kvality. Měření jsou vázána na charakteristiky (subcharakteristiky) a atributy softwarového produktu. Jednotlivým atributům odpovídají míry, měřící funkce, elementy měření kvality a měřící metody.

Míry kvality softwaru jsou voleny tak, aby byly uspokojeny potřeby vývojářů, uživatelů, manažerů a dalších, kteří s informacemi pracují.

Míry se opět dělí do tří kategorií a to tak, aby odpovídaly dříve definovaným pohledům na kvalitu. Jedná se o měření kvality použití, měření vnější a vnitřní kvality. Toto rozdělení mimo jiné koresponduje s životním cyklem kvality softwarového produktu. Viz obrázek.



Obrázek 4: Jakost a životní cyklus produktu. Zdroj: (Vaníček, 2006)

Z obrázku je patrné, že v rámci cyklu postupně vznikají požadavky na kvalitu, následně požadavky na vnější kvalitu a požadavky na vnitřní kvalitu. Měření se definuje také v tomto pořadí, ale vyhodnocení se provádí nejprve na úrovni vnitřní kvality. Toto měření se provádí během vývojových fází, a pokud je správně definováno i prováděno, pak minimalizuje následnou pracnost.

Ve fázi vyhodnocení vnější kvality se vyhodnocuje chování na úrovni logických celků, což jsou části softwarového produktu. V poslední fázi životního cyklu jsou prováděna měření, která odpovídají potřebám specifických uživatelů s ohledem k naplnění jejich business cílů.

V průběhu celého životního cyklu produktu jsou stanoveny míry kvality tak, aby podporovaly úspěšné řízení vývoje, vyhodnocení a podpory softwarového produktu. (Vaníček, 2010)

3.1.7.4 Charakteristiky modelu kvality dat a jejich míry

Charakteristiky definované v modelu kvality dat jsou specifické tím, že se dále nerozpadají na podcharakteristiky a atributy, jako v jiných modelech kvality dat. Proto v následujícím výčtu jsou popsány pouze charakteristiky.

Pod popisem jednotlivých charakteristik je ještě zmíněna míra/míry, které je možno pro charakteristiku použít při jejich hodnocení.

Přesnost

Data správně reprezentují opravdovou hodnotu vloženého atributu.

Přesnost má dva aspekty:

a) Tak zvaná **syntaktická přesnost** definuje blízkost datové hodnoty k množině definovaných hodnot v doméně, které považujeme za správné. Například nesprávně uložená hodnota pro křestní jméno je „Vašek“, korektní hodnota je Václav.

Míra syntaktické přesnosti pro hodnotu atributu záznamu	
Funkce měření	A/B
Měřené elementy kvality	A = počet záznamů, které mají daný atribut syntakticky přesný
	B = počet všech záznamů

Tabulka 2: Ukázka míry syntaktické přesnosti. Zdroj: (Vaníček, 2010)

b) Tak zvaná **sémantická přesnost** definuje blízkost datové hodnoty k definovaným hodnotám v doméně, které považujeme za sémanticky správné. Tím je myšleno, že uložená hodnota má jednoznačně srozumitelný význam. Rozdíl oproti syntaktické přesnosti je v tom, že hodnota sémanticky nepřesná může přesto být syntakticky přesná. Jedná se zejména o hodnoty nebo celé záznamy, které se v rámci systému opakují.

Míra sémantické přesnosti pro hodnotu atributu záznamu	
Funkce měření	A/B
Měřené elementy kvality	A = počet záznamů, které mají daný atribut sémanticky přesný
	B = počet všech záznamů

Tabulka 3: Ukázka míry sémantické přesnosti. Zdroj: (Vaníček, 2010)

Úplnost

Datový subjekt asociovaný s reálnou entitou má vyplněny všechny očekávané atributy a odpovídá reálné entitě v daném kontextu použití.

Míra úplnosti pro data v souboru	
Funkce měření	A/B
Měřené elementy kvality	A = počet záznamů se všemi vyplněnými atributy
	B = počet všech záznamů

Tabulka 4: Ukázka míry úplnosti. Zdroj: (Vaníček, 2010)

Konzistentnost

Data jsou bez rozporů a v souladu s ostatními daty v daném kontextu použití.

Míra konzistentnosti hodnot v datovém souboru	
Funkce měření	A/B
Měřené elementy kvality	A = počet konzistentních záznamů v souboru
	B = počet všech záznamů

Tabulka 5: Ukázka míry konzistentnosti. Zdroj: (Vaníček, 2010)

Důvěryhodnost

Data uživatel považuje za pravdivá a věrohodná v daném kontextu použití.

Míra důvěryhodnosti pro údaje úvěrového rizika	
Funkce měření	A/B
Měřené elementy kvality	A = počet záznamů, které byly označeny vnitřním auditem za důvěryhodné
	B = počet údajů všech úvěrových rizik

Tabulka 6: Ukázka míry důvěryhodnosti. Zdroj: (Vaníček, 2010)

Aktuálnost

Data jsou aktuální v daném kontextu použití.

Míra aktuálnosti pro hodnotu atributu záznamu	
Funkce měření	A/B
Měřené elementy kvality	A = počet záznamů ohodnocených jako dostatečně aktuální
	B = počet všech hodnocených záznamů

Tabulka 7: Ukázka míry aktuálnosti. Zdroj: (Vaníček, 2010)

Přístupnost

Data jsou dostupná v daném kontextu použití, zejména pak pro lidi, kteří potřebují podpůrné technologie nebo zvláštní konfiguraci z důvodu jejich postižení.

Míra přístupnosti pro datové soubory typu zvuk	
Funkce měření	A/B
Měřené elementy kvality	A = počet dat uložených pouze jako „zvuk“ (tj. bez textové reprezentace zvuku)
	B = počet dat reprezentující zvuk

Tabulka 8: Ukázka míry přístupnosti. Zdroj: (Vaníček, 2010)

Shoda s předpisy

Data dodržují normy, konvence, platné předpisy nebo jiná podobná pravidla týkající se kvality dat v daném kontextu použití.

Míra pro soukromí dle zákona: hodnoty záznamů	
Funkce měření	A
Měřené elementy kvality	A = NCP
	NCP = počet položek, které nejsou v souladu se zákonem pro ochranu osobních údajů

Tabulka 9: Ukázka míry pro shodu s předpisy. Zdroj: (Vaníček, 2010)

Důvěrnost

Data jsou přístupná a interpretovatelná pouze autorizovanému uživateli v daném kontextu použití.

Míra důvěrnosti pro použití šifrování	
Funkce měření	A/B
Měřené elementy kvality	A = počet zašifrovaných polí v databázi
	B = počet polí v databázi, které by měly být zašifrovány

Tabulka 10: Ukázka míry důvěrnosti. Zdroj: (Vaníček, 2010)

Výkonnost

Data mohou být zpracována. Dále poskytují očekávanou úroveň výkonu při užití konkrétního množství zdrojů a s konkrétními typy těchto zdrojů. Výkon je stanoven v určitých podmínkách a v daném kontextu užití.

Míra pro zbytečně zabrané místo	
Funkce měření	∑(A-max(B))
Měřené elementy kvality	A = benchmarkově průměrný prostor pro efektivní uložení dat jednoho souboru
	B = maximum použitého místa pro jakýkoliv soubor na fyzickém disku

Tabulka 11: Ukázka míry výkonnosti. Zdroj: (Vaníček, 2010)

Preciznost

Preciznost dat, neboli rozlišení, se kterou je hodnota zaznamenána v daném kontextu použití.

Míra preciznosti hodnoty dat	
Funkce měření	A/B
Měřené elementy kvality	A = počet dat s hodnotou s požadovanou precizností
	B = počet všech hodnot dat

Tabulka 12: Ukázka míry preciznosti. Zdroj: (Vaníček, 2010)

Sledovatelnost

Data poskytují přístup k auditním záznamům a je možné dohledat změny provedené v těchto datech v daném kontextu použití.

Míra sledovatelnosti hodnot	
Funkce měření	A/B
Měřené elementy kvality	A = TVA
	TVA = počet dat, pro které jsou požadované trasované záznamy dostupné
	B = celkový počet dat, které jsou pro trasovatelnost testovány

Tabulka 13: Ukázka míry sledovatelnosti. Zdroj: (Vaníček, 2010)

Srozumitelnost

Data je možno přečíst a interpretovat uživatelům v příslušném jazyce, symbolech nebo jednotkách v daném kontextu použití.

Míra srozumitelnosti vzhledem k existujícím metadatům	
Funkce měření	A/B
Měřené elementy kvality	A = počet dat s existujícími metadaty
	B = počet dat hlavního datového souboru

Tabulka 14: Ukázka míry srozumitelnosti. Zdroj: (Vaníček, 2010)

Dostupnost

Data jsou dostupná pro autorizované uživatele v daném kontextu použití.

Míra stupně dostupnost položek	
Funkce měření	A/B
Měřené elementy kvality	A = počet datových položek dostupných během zálohy/obnovy
	B = počet datových položek datové zálohy/obnovy

Tabulka 15: Ukázka míry dostupnosti. Zdroj: (Vaníček, 2010)

Přenositelnost

Data mohou být přesunuta z jedné platformy na jinou při zachování stávající kvality v daném kontextu použití.

Míra přenositelnosti	
Funkce měření	A/B
Měřené elementy kvality	A = počet dat, které si zachovali stávající kvalitu po přechodu do jiného počítačového systému
	B = počet migrovaných dat

Tabulka 16: Ukázka míry přenositelnosti. Zdroj: (Vaníček, 2010)

Obnovitelnost

Data jsou obnovitelná a po obnovení udržitelná a je možné na nich provést stupeň operací v dané kvalitě a to i v případě selhání v daném kontextu použití.

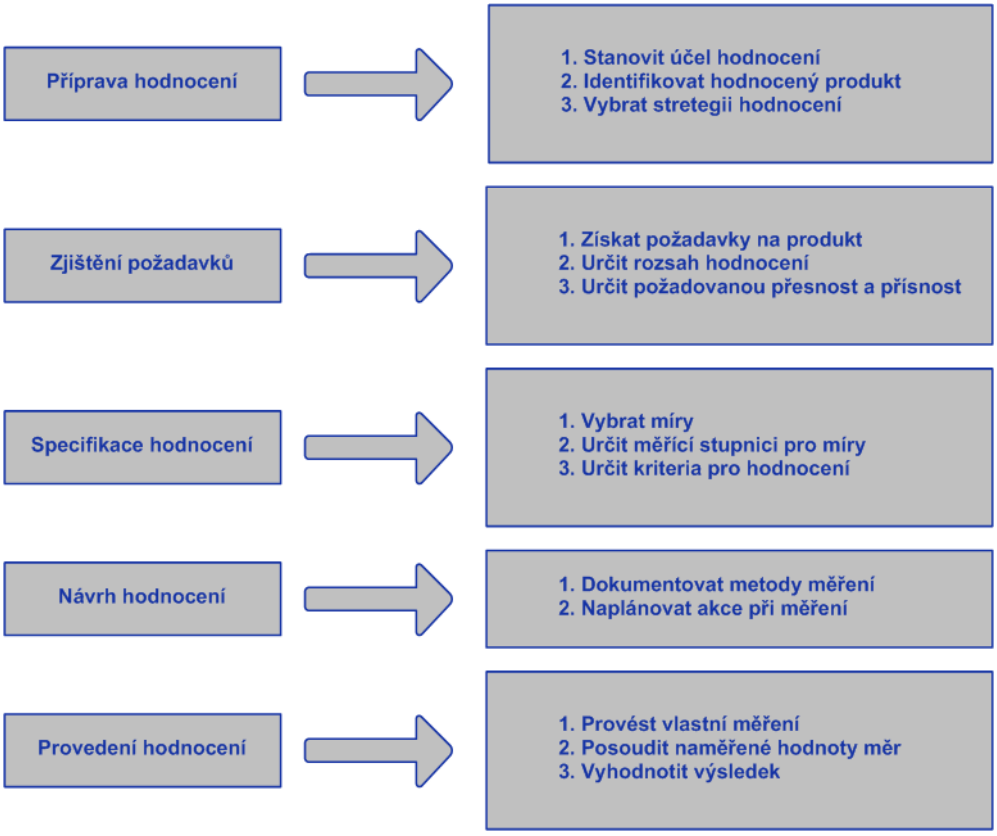
Míra stupně obnovitelnosti	
Funkce měření	A/B
Měřené elementy kvality	A = počet položek dat úspěšně zálohovaných/obnovených
	B = počet zálohovaných/obnovených položek dat

Tabulka 17: Ukázka míry obnovitelnosti. Zdroj: (Vaníček, 2010)

3.1.7.5 Postup vyhodnocení kvality dat

V předchozích kapitolách byly zmíněny charakteristiky kvality dat, ale aby bylo možné s těmito hodnotami pracovat, musí být nejprve získány, zpracovány a na závěr vyhodnoceny. Tento proces má v rámci metodiky SQuaRE konkrétní etapy. (Vaníček, 2010)

Viz obrázek.



Obrázek 5: Etapy při hodnocení jakosti. Zdroj: (Vaníček, 2006)

Výsledkem tohoto procesu je výstup o kvalitě dat.

3.1.7.5.1 Příprava hodnocení

Jedná se o první fázi, kdy získáváme prvotní informace o objektu zkoumání. Připravuje se identifikace hodnocené entity. Stanovujeme účel hodnocení, proč a k čemu hodnocení bude použito. Zjišťují se informace o potřebách uživatelů, životním cyklu produktu atd. Dále se v této etapě určuje strategie hodnocení.

3.1.7.5.2 Zjištění požadavků

Etapa zjištění požadavků se skládá ze tří částí.

- Za prvé se stanoví požadavky na jakost. Jsou vybrány určité atributy jakosti, které budou měřeny. Dále pro tyto atributy jsou zvoleny také způsoby hodnocení.
- Za druhé je určen rozsah pokrytí požadavků, který je stanoven na základě rozpočtu a velikosti rozsahu.
- Za třetí stanovujeme přesnost a přísnost hodnocení, které opět závisí na konkrétních potřebách uživatele.

3.1.7.5.3 Specifikace hodnocení

V této etapě se způsob měření kvality připravuje detailně. To znamená, že se vybírají užité míry vzhledem ke zvoleným atributům jakosti. Dále se určují měřící stupnice a na závěr této etapy se určí kritéria pro hodnocení. Stanovení kritérií pro hodnocení je velmi důležité, protože na základě hodnocení se stanovují výsledky kvality softwarového produktu.

3.1.7.5.4 Návrh hodnocení

Během této etapy se má vytvořit podrobný plán, který by již během samotného měření a vyhodnocování neměl být měněn. V případě změny během měření by mohlo dojít ke ztrátě objektivnosti. V rámci této etapy se dále musí dokumentovat metody měření a také stanovit plán akcí při měření, aby nedocházelo k opakování činností, ale na druhou stranu žádná činnost nebyla vynechána.

3.1.7.5.5 Provedení hodnocení

Vlastní proces hodnocení probíhá již podle schváleného návrhu. Ten byl dohodnut mezi stranou, která hodnocení provádí a stranou, která hodnocení financuje. Konkrétně hodnocení probíhá následovně:

a) Získání hodnot měr – pozorováním nebo dopočtem v případě odvozených měr
b) Posouzení hodnot měr
c) Zhodnocení výsledků

Na úplný závěr jsou vyhodnoceny jednotlivé dílčí výsledky a následně podle stanovených funkcí a vah jsou sumarizovány pro jednotlivé podcharakteristiky a poté pro charakteristiky. Výsledné hodnoty jsou porovnány se stanovenými požadavky a úplným výsledkem je odpověď na položenou otázku. Tato otázka byla stanovena, když se určoval účel daného měření. Například se může jednat o konečnou akceptaci daného softwarového produktu. (Vaníček, 2006)

3.1.8 Prostředky pro podporu kvality dat v databázích

V databázích je integrována celá řada prostředků, které při správném použití vyžadují od uživatele dodržování kvality dat. Data „automaticky“ kontrolovaná jsou následně kvalitnější. Dále jsou popsány prostředky, které zvyšují kvalitu dat, některé jsou typické pro databáze, některé se vyskytují napříč IT technologiemi.

3.1.8.1 Deklarace velikosti a datových typů proměnných

Správně definovaný typ proměnné je základním kamenem pro ukládané informace. Protože přes název proměnné identifikujeme, o jakou informaci se jedná, v terminologii názvu může být zakotven i typ proměnné pro rychlou orientaci. Proto jednoznačné názvy a jmenné konvence jsou naprostým základem pro jednoznačnost a orientaci v informacích. Správně použitý datový typ dále specifikuje, o jaká data se bude jednat, zda měsíc, číslo, interval atd. Datový typ také slouží i pro jednoduchou kontrolu formátu, například není možné do datového typu datum uložit třináctý měsíc. Proto není dobré používat univerzálně pro všechna data datový formát řetězec.

Zmíněným ukazatelem kvality, který s deklarováním proměnných souvisí, je preciznost. Tato vlastnost se týká především ukládání číselných hodnot a jejich zaokrouhlováním, proto je potřeba zvolit správné rozlišení a správně zvážit velikost zaokrouhlení, abychom ukládali data, která budou vypovídající o reálném objektu. (Elliott, 2015)

3.1.8.2 Integritní omezení

Tato omezení jsou více technického charakteru, ale významně se podílejí na dodržování struktury a validity dat. Jedná se o řešení týkající se relačních databází.

Jedná se o databázové prostředky, které určují vlastnosti ukládaných informací tak, aby byly úplné, kompletní i validní. Proto se dá říci, že se starají o dodržování požadovaných pravidel a tak jsou další částí, které obstarávají kvalitu dat.

Prvním z integritních omezení je referenční integrita, která se skládá z primárních a cizích klíčů. Primární klíče slouží k identifikaci a k jednoznačnému určení jednotlivých řádků v tabulce. Musí to být unikátní hodnota. Tou může být například jedinečný identifikátor, např. číslo nebo kombinace několika sloupců, které jsou jedinečné a identifikují danou entitu. Primární klíč je kombinací integritních omezení not null a unique. Cizí klíč slouží k propojení dvou tabulek. Defacto se jedná o vložení primárního klíče jedné tabulky do sloupce druhé tabulky, čímž se zabezpečí požadovaná provázanost. (Harrington, 2010)

Druhým z integritních omezení je check constraint, který aplikuje na úrovni sloupce v tabulce určité pravidlo, které pokud není splněno, je vyvolána výjimka a není možné danou hodnotu do sloupce vložit. Může se jednat o jednoduché pravidlo, kdy daná hodnota nesmí být záporná, například cena za určitou surovinu, nebo složitější pravidlo, kdy daná hodnota musí splnit více podmínek. (Harrington, 2010)

Třetím z integritních omezení je unique constraint. Tento constraint byl již zmíněn v prvním odstavci o referenčních integritách a je součástí primárního klíče. Jeho omezením je to, že tato hodnota v rámci sloupce tabulky musí být jedinečná a nesmí se v žádném jiném řádku opakovat. Příkladem může být katalog výrobků, kde by se žádný výrobek neměl opakovat, všechny by měly být právě jednou. (Harrington, 2010)

Zmiňované integritní omezení výrazně snižují chyby v uložených datech, protože uložit nesmyslná data prostě nedovolí. Bohužel někdy je potřeba tato omezení dočasně vypnout a pak vzniká prostor pro vznik chyb. Důvody mohou být různé, například výkonnostní, pak je ale na zvážení, zda je žádoucí podstoupit toto riziko.

3.1.8.3 Transakčnost a konzistentnost

Dalšími podpůrnými funkčnostmi pro kvalitu dat je transakčnost a konzistentnost. Tyto dvě vlastnosti spolu úzce souvisí a zabezpečují, aby informace v databázi byla zaznamenána vždy kompletně na konci transakce. Konzistentnost znamená, že data jsou v celém rozsahu kompletní. V žádném místě není uložena částečná informace, která v kontextu ostatních dat nedává smysl. Transakčnost znamená, že data jsou měněna v rámci transakcí na konci sledu několika příkazů.

Databáze podporují ještě celou řadu funkčností, které zabezpečují kvalitu dat. Jedná se o zálohování, archivaci, autentizaci, autorizaci a mnoho dalšího. V konečném důsledku závisí na tom, jak je využit potenciál všech těchto možností, aby byla zachována maximální možná míra kvality dat. (O'Neil, 1994)

Co se týká ukládání dat v databázi, je důležité zmínit, že je potřeba nejen ukládat správná data, ale zároveň je ukládat i správně a na správná místa, aby v konečném důsledku byla data nejen kvalitní, ale navíc pro uživatele i použitelná, dohledatelná atd. Tato oblast se týká více metodik a jejich dodržování, ale s kvalitou dat také souvisí.

3.2 Čištění dat a deduplikace

Jak bylo zmíněno v úvodu, je ze strany businessu velice žádoucí, aby informační systém jednak podporoval chod firmy a také byl řízen chodem firmy. Navíc je i zapotřebí, aby se přizpůsoboval měnícím se požadavkům, které by měly reagovat na aktuální vývoj trhu. Toto platí nejen pro informační systém, ale i pro data a kvalitu dat. Navíc v datech nejsou jen informace pro aktuální řízení organizace, ale i znalostní databáze všech minulých obchodů, transakcí a nabídek. Tato data slouží pro podporu obchodu a to tak, že dávají nepřímou zpětnou vazbu. Aby uložené informace mohly sloužit ke všem vyjmenovaným účelům, musí být kvalitní.

Toto je jeden z důvodů důležitosti kvality dat. Dalším důvodem, proč se v rámci firmy snažit o kvalitu dat, je snadnější administrace, spravování, zálohování i aplikační podpora. Z pohledu IT, pokud by nebylo prováděno kontinuální čištění dat, pak by data mohla narůst například každý rok o deset procent nad standardní přírůstek uživatelských dat. Může se zdát, že tento přírůstek je malý, ale dodatečné náklady na jeho správu jsou nezanedbatelné. V tomto pohledu jsem bral v úvahu pouze náklady na IT. Dalšími náklady by byly ušlé zisky při kontaktování nereálných klientů apod. Souhrnně se tato oblast nazývá cena za kvalitu nebo také „Náklady na kvalitu (Cost of quality).

3.2.1 Náklady na kvalitu

V rozsáhlých softwarových systémech, které obsahují komplexní složité procesy, je následně náročná jejich údržba. Proto není překvapující, že právě náklady na údržbu a vývoj tvoří 80 procent celkových nákladů na software. Dalším faktem, který toto pouze doplňuje, je to, že programátoři stráví okolo 60 procent času pouze porozumění kódu daného systému. (Boehm, 2001)

Není žádných pochyb, že kvalita softwaru stojí čas a peníze. Jakýkoliv nedostatek v kvalitě pak vytváří dodatečné náklady jak na straně uživatelů, tak na straně programátorů. Proto je žádoucí, aby většina chyb byla odstraněna v rámci vývoje funkčnosti, protože stejná chyba objevená v rámci vývojových testů stojí desetkrát méně, než stejná chyba objevená v rámci předávání uživateli. (Boehm, 2001)

Cost of quality neboli náklady na kvalitu jsou abstraktním souhrnem všech nákladů, které jsou spojeny s kvalitou i zajišťováním kvality dat. Jsou zde zahrnuty náklady na vývojový tým, jeho činnosti spojené s podporou a nápravou chyb prokazatelně spojených s nedostatečnou kvalitou. Tyto náklady jsou pak rozděleny mezi tyto činnosti: preventivní aktivity, vyhodnocující aktivity a opravy selhání.

Náklady na preventivní činnosti pokrývají aktivity zaměřené na prevenci proti špatné kvalitě produktu a služeb (např. plánování kvality, zlepšování kvality projektu). Náklady na prevenci by měly také obsahovat ty náklady, které jsou vynaloženy na zaškolování týkající se kvality dat.

Náklady na vyhodnocení pokrývají aktivity spojené s měřením, vyhodnocováním nebo auditem produktů a služeb v rámci zajištění standardů kvality a požadavků na kvalitu.

Posledním typem nákladů, který je zohledněn v rámci nákladů na kvalitu, jsou náklady na selhání. Tyto náklady jsou rezervovány pro nápravu nedostatečné kvality produktu/služeb, která vyplývá z požadavků/potřeb zákazníka. Tyto náklady pak pokrývají tyto činnosti: podporu, opravy nebo náhradu produktu v případě větších nedostatků. Tento rozpočet může také zohledňovat ušlé náklady v případě ztráty důvěry nebo zhoršení pověsti společnosti. (Soley, 2016)

3.2.2 Udržování kvality dat

V předchozím odstavci bylo popsáno, jak je důležitá kvalita dat v informačních systémech, a že náklady navíc při zanedbání kvality nejsou zanedbatelné. Proto vzniká požadavek na to, aby informace zůstala kvalitní i v průběhu času. To je relevantní požadavek na to, aby se s informací pracovalo tak, aby se aktualizovala, očišťovala od chyb i neaktuálních informací. A stále musí nést takovou informovanost, jako při prvotním uložení. Informace samozřejmě nemusí být kvalitní už při prvotním uložení. O toto by se měla starat různá validační pravidla a logika. Tato pravidla mohou být aplikována na různých vrstvách systému, nejčastěji se však objevují na úrovni front-endu, nebo mohou být prováděna

během zpracování. Další možností je aplikovat tato pravidla až při ukládání na perzistentní úložiště a při selhání pokusu o uložení vrátit chybu uživateli na front-end. Pravidla mohou být různých typů, mohou ověřovat správný formát dat, správnou návaznost dat nebo strukturu, smysluplnost dat a další. Tímto se dostáváme k oblasti validace dat. V případě, že validace dat neproběhla, musí být provedeno čištění dat, aby data obsahovala kvalitní informaci.

Zde dále už bude popsáno pouze čištění na úrovni databází, což je součástí procesů kontroly a udržování kvality dat, pro toto zpracování není nutné data prezentovat/zobrazovat na front-endu. Jedná se už pouze o určitou formu dat, ale je důležité, aby data měla požadované vlastnosti a ne, aby byla konkrétním způsobem prezentována.

Čištění dat má v procesu udržování kvality dat důležitou pozici. Odpovědnost za kvalitu dat v rámci firmy má na starosti samostatný obor – Quality management. Tento obor má za úkol obecně zajišťovat kvalitu procesů a potažmo produktů v organizaci, mimo jiné i kvalitu dat.

3.2.3 Problémy s kvalitou – typy chyb

Problémy s kvalitou (Quality Problems) nemají význam pouze ve smyslu chyb v datech, jsou to i nekonzistence, významové nesmysly atd. Podle Chapmena (Chapman, 2005) by v chybách a nekvalitních datech neměl být viděn jen problém, je to i zpětná vazba a prostor ke zlepšení. Pokud o chybě víme, je zde prostor ji opravit. Pokud o chybě nevíme, neznamená to, že v systému neexistuje, ale s největší pravděpodobností pouze ještě nebyla odhalena.

Podle Erharda Rahma (Rahm, 2000) jsou problémy s kvalitou děleny na nedostatky z jednoho zdroje a nedostatky (problémy) způsobené z více různých zdrojů. Dále se problémy dělí na problémy na úrovni schématu, které reflektují chyby na úrovni designu a na problémy na úrovni instance, které odkazují na chyby a nekonzistence jednotlivých aktuálních záznamů, které nejsou rozpoznatelné na úrovni schématu.

Obrázek 6: Přehled možných problémů s kvalitou. Zdroj: (Rahm Erhard, 2000)

3.2.3.1 Chyby jednoho zdroje dat

Kvalita dat daného zdroje velmi závisí na zavedených schématech a integritních omezeních, které tento zdroj dat popisují a vymezují. Tato schémata zároveň definují dovolené vstupní hodnoty. U zdrojů bez schématu (např. soubor dat), kde není velké množství omezení a které lze snadno uložit, mohou velmi jednoduše vznikat chyby a nekonzistence. U takových zdrojů dat pak pouze závisí na lidském faktoru, zda chyba vznikne nebo nevznikne, ale je spíše otázkou času a nepozornosti uživatelů, než je nějaká chyba zavlečena do datového souboru. Proto vznik chyb u jednotlivých souborů dat je velice jednoduchý.

Databázové systémy na druhou stranu zajišťují velké množství restrikcí specifického datového modelu (relační přístup, jednoduché hodnoty atributů – atomicita, referenční integrita, řízení přístupu a další). Proto nedostatky zapříčiňující vznik chyb a nekonzistencí v datech těchto systémů jsou způsobeny hlavně nedostatečnou nebo nepřesnou specifikací jednotlivých pravidel, a potažmo datového modelu, nebo nedostatečných popisujících schémat, nebo protože pouze některá integritní omezení byla nedefinována.

Nedostatky vzniklé špatným definováním schémat (entitních diagramů) způsobují chyby typu schema level problem, neboli problémy na úrovni schématu. Tyto chyby mohou například vzniknout, když místo typu datum ukládáme datum jako řetězec a následně jej v lepším případě převádíme na datový typ datum a v horším případě ho i jako řetězec

reprezentujeme. Pak mohou vznikat nesmysly jako například 13. měsíc apod. Typy problémů na úrovni schémat jsou v následující tabulce:

Problém		Špatné záznamy	Důvody/Označení
Atribut	Nepřípustná hodnota	Datum narození = 30.13.70	Měsíc mimo povolený rozsah
Záznam	Porušená návaznost atributů	Věk = 22, datum narození = 12.02.70	Věk = (dnešní datum – datum narození) se má rovnat
Typ záznamu	Porušena unikátnost	Emp1=(jméno="Jan Novák", ID ="007") Emp2=(jméno="Petr Novák", ID ="007")	Unikátnost pro ID (identifikátor záznamu) byl porušen
Zdroj	Porušena referenční integrita	Emp(jméno="Jan Novák", oddělení = 255)	Reference na oddělení (255) není definována, oddělení neexistuje

Tabulka 18: Příklady chyb z jednoho zdroje dat na úrovni schématu. Jsou porušena integritní omezení. Zdroj: (Rahm Erhard, 2000)

Problém		Špatné záznamy	Důvod/Označení
Atribut	Chybějící hodnota	Telefon = 9999-999999	Nereálné hodnoty, během vkládání dat (dummy hodnoty nebo prázdné)
	Pravopisné chyby	Město = "Plyeň"	Obvykle překlepy, fonetické chyby
	Zkratky	Zažito = „DB" povolání = „DB programátor"	
	Vložená hodnota	Jméno = „J. Novák 12.02.70 Praha"	Více hodnot vloženo do jednoho atributu
	Nesmyslná hodnota	Stát="Praha"	
Záznam	Nepřípustná hodnota při závislosti	Město = „Ostrava" PSČ = "263 01"	Město a PSČ neodpovídají
Typ záznamu	Transpozice slov	Jméno1 = „J. Novák" jméno2 = „Vomáčka M."	Obvykle ve volných polích, vznikají různá pořadí atributů
	Duplikující hodnoty	Emp1 = (jméno = "Jan Novák"), Emp2 = (jméno = „J. Novák")	Stejní zaměstnanci jsou uloženi jako dvě rozdílné osoby
	Odporující záznamy	Emp1 = („Jan Novák", nar.= 12.02.70) Emp2 = („Jan Novák", nar. 12.12.70)	Stejný uživatel je popsán různými hodnotami v atributech
Zdroj	Špatné odkazy	Emp= (jméno = „Jan Novák", oddělení = 17)	Odkaz na oddělení je reálný, ale chybný. Oddělení je špatně přiřazeno.

Tabulka 19: Příklady chyb z jednoho zdroje dat na úrovni instance. Zdroj: (Rahm Erhard, 2000)

Problémy specifické pro instanci jsou chyby nebo nekonzistence, které nelze definovat pomocí schémat, například překlepy. Jsou to chyby, které nelze zachytit pomocí

integritních omezení, jedná se zejména o překlepy nebo nesprávné/zavádějící/nesmyslné hodnoty vložené uživatelem do vstupních formulářů.

3.2.3.2 Chyby zapříčiněné různými zdroji dat

Chyby vzniklé v jednotlivých zdrojích dat jsou znásobeny/zhoršeny v případě slučování více zdrojů dat do jediného. Každý zdroj dat může obsahovat nekvalitní data a navíc data v různých zdrojích mohou být reprezentována odlišně, mohou se překrývat nebo být v rozporu. Tyto nekonzistence vznikají na základě toho, že každý zdroj dat je nezávisle vyvíjen a udržován vzhledem ke specifickým požadavkům od zadavatelů. To má za následek značné rozdíly v jednotlivých datových modelech, schématech i skutečných datech. Hlavní příčinou vzniku rozdílů jsou rozdílné jmenné konvence, kdy buď používáme stejné názvy pro různé objekty – homonyma, nebo používáme různé názvy pro stejné objekty – synonyma.

Dalším typem chyb na úrovni schématu jsou chyby zapříčiněné různými reprezentacemi v různých datových zdrojích, např. duplikované záznamy, protichůdné záznamy, negované záznamy atd. V těchto případech mohou existovat stejné názvy atributů i datové typy, ale reprezentace je rozdílná, např. hodnota pro rodinný stav, nebo hodnota pro uloženou měnu (euro versus dolar). Chybou v reprezentaci je i rozdílný stupeň uložených agregovaných záznamů, např. tržby za produkt versus tržby za skupinu produktů atd.

Hlavním problémem pro čištění dat vzniklých z více zdrojů je identifikace překrývajících se dat, které se částečně shodují, respektive odkazují na stejnou entitu, například zákazníka. Tento problém je také označován jako problém identity objektu (entity), zároveň se také používá označení eliminace nebo slučování/deduplikování.

Často jsou informace z více zdrojů pouze částečně duplikované a jednotlivé zdroje dat se mohou vzájemně doplňovat a poskytovat tak další informace o dané entitě. Samozřejmě duplikované informace by měly být očištěny a doplňující informace by měly být konsolidovány a sloučeny za účelem dosažení co nejvíce reálného pohledu na danou entitu v reálném světě.

Nejnázorněji jsou problémy se slučováním dat z různých zdrojů zřetelné na konkrétním příkladu. V následujících prvních dvou tabulkách jsou data z dvou různých zdrojů. V třetí tabulce jsou záznamy sloučené z prvních dvou tabulek.

ID	Jméno	Ulice	Město	Pohlaví
11	Marie Nováková	Strakonická 10	Praha	0
24	Petr Dohnal	Plzeňská 125	Karlovy Vary	1

Tabulka 20: Zákazníci – 1. zdroj dat. Zdroj: (Rahm Erhard, 2000)

Číslo ID	Příjmení	Jméno	Pohlaví	Adresa	Telefon
10	Nováková	Marie	F	Strakonická 10, Praha	736 456 958
25	Dohnal	Petr	M	Lidická 12, Praha	735 256 456

Tabulka 21: Klienti – 2. zdroj dat. Zdroj : (Rahm Erhard, 2000)

Č.	Příjmení	Jméno	Pohlaví	Ulice	Město	Telefon	ID	Číslo ID
1	Nováková	Marie	F	Strakonická 10	Praha	736 456 958	11	10
2	Dohnal	Petr	M	Plzeňská 125	Karlovy Vary		24	
3	Dohnal	Petr	M	Lidická 12	Praha	735 256 456		25

Tabulka 22: Příklad ukázky vzniku chyby při sloučení záznamů z různých zdrojů dat, jedná se o chyby na úrovni schématu, tak na úrovni instance. Zdroj: (Rahm Erhard, 2000)

Z ukázek je patrné, jak k duplikaci došlo. Tento vzniklý stav musí být zvážen a na základě ještě dalších proměnných musí být rozhodnuto, zda se jedná o dvě rozdílné osoby anebo jednu a tu samou.

3.2.4 Nekvalitní data (Dirty data)

Dirty data je termín, který není jednoduché správným způsobem přeložit. Jedná se o jakási špinavá data, nefinální data, nekvalitní data. Tento termín se používá v IT pro označení nesprávných/nepřesných dat shromážděných v rámci různých formulářů. Někdy je tento pojem také použit pro označení dat, která nejsou v databázi ještě potvrzená, tzv. „commitnutá“, a jsou uchovávána zatím pouze v paměti.

Dirty data („špinavá data“) mohou být zavádějící, nesprávná, bez správného formátování, nesprávně hláskovaná nebo se špatnou interpunkcí. Mohou být vložena do špatného pole nebo i duplikovaná. Abychom výskyt těchto dat minimalizovali, je vhodné použít validace.

Jsou různé příčiny vzniku dirty dat. V některých případech je informace záměrně zkreslena. Někdy může uživatel vložit zavádějící nebo fiktivní osobní informace, které

se zdají reálné. Takový typ dat není možné jednoduše rozpoznat nebo validovat rutinními procesy, protože se takový záznam zdá být správný.

Dalším typem dirty dat jsou data duplikovaná. Tato data mohla vzniknout opakovaným odesláním dotazníků nebo chybným spojením různých zdrojů dat. V těchto typech dat mohou být chyby ve formátování nebo typografické chyby. Běžnou chybou uživatelů je různá preference formátování telefonních čísel. (Adelman, 2005)

Dirty data je dalším pohledem na nesprávná nebo chcete-li chybová data, ale spíše se jedná o data dosud nezpracovaná nebo předpřipravená a na zpracování čekající. Každopádně jsou tato data bez dalších úprav nepoužitelná. Nepoužitelná ve smyslu pro koncového uživatele.

3.2.5 Postup čištění dat

Postup čištění dat by měl splňovat několik požadavků. V první řadě by měl detekovat a odstraňovat hlavní chyby a nekonzistence nezávisle na individuálnosti dat a integraci více datových zdrojů. Dále by přístup měl podporovat nástroj, který bude označovat data k manuální uživatelské kontrole, ale toto by mělo nastávat v minimálním množství případů, aby bylo minimalizováno dodatečné úsilí kontroly. (Rahm, 2000)

Konkrétně by podle Erharda Rahma (Rahm, 2000) postup čištění měl obsahovat tyto body:

a) **Analýza dat** – tento krok se provádí pro zjištění, jaké druhy chyb a nesrovnalostí mají být opraveny. Analýza obsahuje i manuální kontrolu vzorku dat. Výsledkem této části by měl být soubor metadat, které budou popisovat vlastnosti dat a detekovat potenciální chyby a problémy s kvalitou, které mají být odstraněny.
b) **Definice transformačního procesu a určení mapovacích pravidel** – v rámci tohoto kroku jsou definovány procesy pro čištění dat, jejich rozsah a množství je závislé na množství datových zdrojů, jejich heterogenitě, stupni kvality i rozsáhlosti dat. V některých případech je potřeba vytvořit transformační schéma a někdy mapování jednotlivých entit. Definice transformačního procesu vychází z metadat z předchozího kroku.
c) **Verifikace** – kontrola správnosti a efektivnosti transformačního procesu a definice transformace. Transformace by měla být testována a ohodnocena. Cíl tohoto kroku je ověřit dosažení požadovaných vlastností pomocí procesu transformace.
d) **Transformace** – provedení jednotlivých kroků transformace začínající načtením a končícím uložením upravených dat.

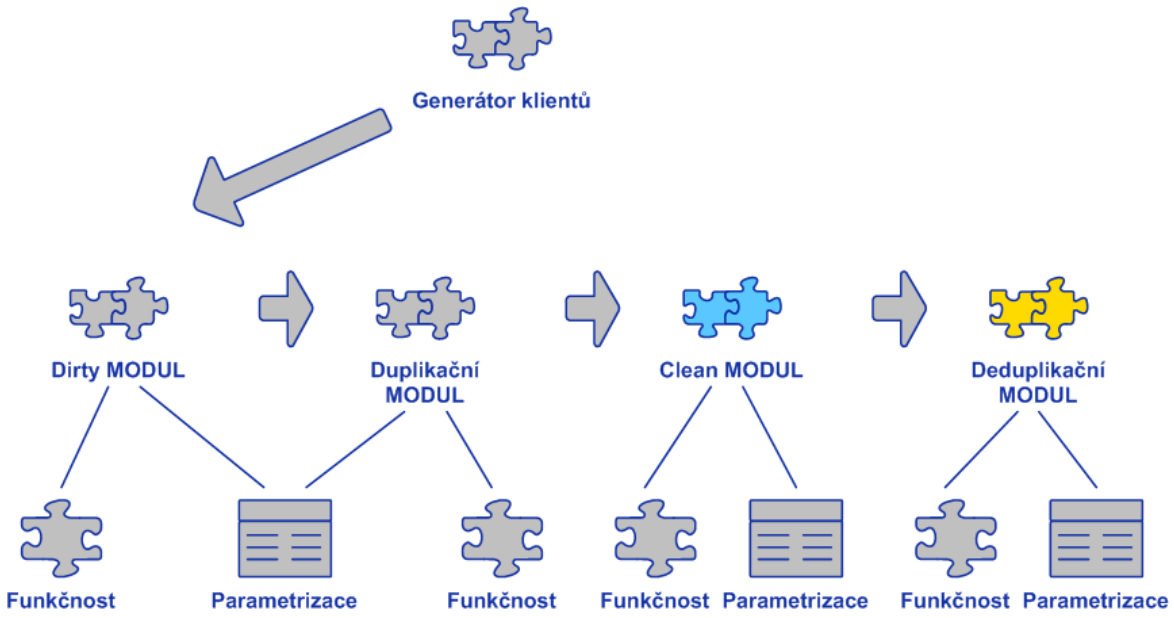
e) **Zpracování očištěných dat** – po odstranění chyb, jsou očištěná data načtena do původních dat a nahrazují původní nekvalitní (dirty) data. Cílem tohoto kroku je poskytnout vylepšené údaje tak, aby nebylo dále více potřeba data aktuálně čistit.

Jak je patrné, nejdůležitějším krokem je správná definice chyb, jejich rozpoznání a definování postupu opravy chybných stavů. Proto je potřeba, aby bylo jasné, jak mají správná data vypadat, a tento stav jednoznačně definovat. Na základě těchto definic jsou pak hledány rozpory a nekonzistence. Nejsložitější je hledání chyb, které jsou vyvolány vzájemnými vztahy více polí. Pro tento typ se musí provádět hlubší analýza a reporty stavů, aby byly tyto chyby odhaleny. (Maletic, 2000)

4 Praktická část

Praktická část popisuje samotný deduplikační skript, který dokáže rozpoznat stejné nebo i podobné záznamy a podle stupně a velikosti vzájemných rozdílů je pak dokáže sloučit. Samozřejmě veškeré změny zaznamenává, aby bylo možné data vrátit do původní podoby. Všechny procesy transformace, ať se jedná o úpravu dat a duplikaci pro simulaci reálných dat, nebo následně samotné čištění a deduplikaci, vyžadují velké množství metadat, parametrizací, datových schémat, definic workflow atd. Je velmi užitečné, aby tato podpůrná data byla konzistentně uložena a dokumentována, aby bylo možné je spravovat a zároveň je opakovaně používat (Rahm, 2000). I pro tyto účely byly vytvořeny podpůrné služby, které slouží jednak k uchovávání parametrizací a jednak zaznamenávání všech dokončených činností.

Na následujícím schématu jsou znázorněny hlavní moduly praktické části:



Obrázek 7: Moduly čistícího skriptu. Zdroj: autor.

Každý z modulů se skládá z vlastní funkčnosti a parametrizace. Někdy může být parametrizace sdílená.

Moduly:

- Generátor uživatelů – generuje náhodné uživatele na základě číselníků, tento modul nebyl implementován v rámci této práce, ale byl převzat z mé předchozí bakalářské práce.

- Upravující modul (Dirty module) – tento modul slouží k náhodnému upravení vygenerovaných dat, může se jednat o jednoduché překlepy nebo i větší chyby v rámci záznamu o uživateli.
- Duplikační modul (Duplicate module) – vytváří v rámci upravených dat kopie stávajících záznamů uživatelů, v rámci těchto duplikovaných dat navíc data také upravuje, pro tyto úpravy jsou využívány části Dirty modulu.
- Čistící modul (Clean module) – tento modul kontroluje a opravuje data. Kontroly jsou prováděny buď proti číselníkům, nebo definovaným pravidlům/vzorům.
- Deduplikační modul (Deduplicate module) – slouží k rozpoznávání stejných nebo podobných záznamů o uživatelích a jejich slučování. Samozřejmě v případě chybného vyhodnocení je možné akci odrolovat na základě záznamů o sloučení.
- Podpůrné služby – podpůrné služby slouží k zaznamenávání proběhlých činností a z těchto záznamů lze podrobně dohledat všechny kroky jednotlivých modulů. Tyto záznamy jsou velmi užitečné, protože velké množství činností modulů je čistě náhodné a neopakovatelné, a z toho důvodu následně těžko napodobitelné pro zjištění příčiny nebo případné chyby.
- Číselníky – slouží ke generování dat a k jejich kontrole.

Číselníky jsou používány jednak k vygenerování dat a jednak ke kontrole dat, zda údaje o klientech odpovídají hodnotám v číselnících. Číselníky byly staženy z internetových stránek Českého statistického úřadu (dále pouze ČSÚ) a následně byly zpracovány tak, aby byly použitelné v rámci skriptů. V rámci práce jsou použity dva „druhy“ číselníků.

Číselník zdrojových dat

Prvním typem jsou data v tabulce source_data, která se používají k prvotnímu vygenerování dat. Tato data nesou informaci o své četnosti. Údaj o četnosti byl stažen z ČSÚ v rámci dalších údajů a odpovídá četnosti jednotlivých typů dat v roce 2011. Jedná se o počet lidí v ČR, kteří tento atribut v daném roce splňovali. Například pro typ dat birth_year pro hodnotu 1986 je uložena četnost 142 281. Toto znamená, že v ČR v roce 2011 bylo 142 281 lidí s rokem narození 1986.

Jednotlivé typy dat včetně jejich dalších informací jsou uloženy v tabulce source_data. Je zde pět typů dat a jsou následující:

- Birth_year – rok narození
- Part_mob_number – první trojčíslí mobilního čísla
- Town – město
- First_name – křestní jméno
- Surname – příjmení

Typ křestní jméno a příjmení nesou navíc informaci, zda se jedná o údaj určený pro ženu nebo pro muže. Tento údaj je uložen ve sloupci type_sex s příznakem M – muž nebo Z – žena.

Ukázka z číselníku:

Hodnota	Četnost	Typ	Pohlaví
ŘEHŮŘEK	280	surname	M
ŘEHŮŘKOVÁ	276	surname	Z
1985	145874	birth_year	
Jistebník	1530	town	
VIOLETA	163	first_name	Z

Tabulka 23: Ukázková data typu zdrojová data. Zdroj: autor.

Číselník adres

Druhým typem dat jsou adresy. Adresy se skládají z města (uložena v tabulce towns), ulice (uloženo v tabulce streets) a čísla popisného (uloženo v tabulce house_numbers). I tato data odpovídají reálným datům v ČR a byla také stažena z ČSÚ. Datová struktura je logická, pro město jsou uvedeny všechny jeho ulice a pro jednu ulici jsou uvedena všechna čísla popisná.

Ukázka z číselníku:

Město	Ulice	Č.P.
KAŠPERSKÉ HORY	AMÁLINO ÚDOLÍ	375
KAŠPERSKÉ HORY	AMÁLINO ÚDOLÍ	397
KAŠPERSKÉ HORY	BAAROVA	132
KAŠPERSKÉ HORY	BARVÍŘSKÁ	77

Tabulka 24: Ukázková data typu adresa. Zdroj: autor.

Oba druhy číselníků byly již zpracovávány a používány v rámci mé bakalářské práce, zde byly jen částečně upraveny a použity i do čistícího skriptu.

4.1 Funkční moduly

Čistící skript se skládá ze čtyř modulů a navíc je zde ještě použit generátor dat, který byl implementován v rámci mé bakalářské práce. Všechny tyto moduly zde budou popsány a na ukázkách kódu bude předvedena jejich funkčnost.

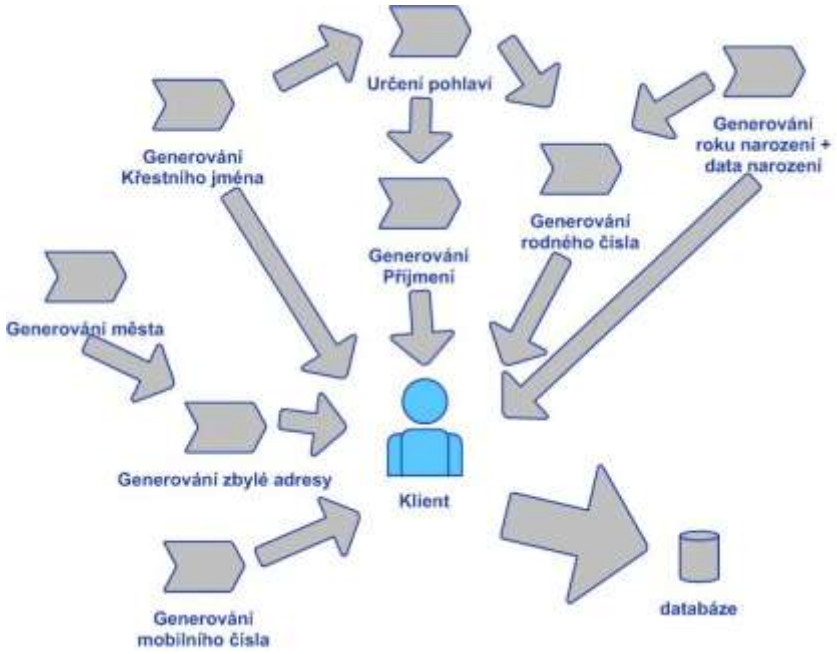
4.1.1 Generátor klientů

Tento modul slouží k vygenerování náhodných uživatelů. Generování je založeno na číselnících, které byly staženy z ČSÚ. Pravděpodobnost vložení jednotlivých údajů do generovaných dat je založena na četnosti výskytu daného údaje. Čím vyšší četnost má daný údaj, tím je pravděpodobnější, že bude vložen generátorem do generovaných dat.

4.1.1.1 Postup generování

V rámci vstupních parametrů je zadán požadovaný počet generovaných klientů a podle tohoto parametru skript vygeneruje daný počet klientů. Dalším parametrem je název tabulky, která bude v rámci generování vytvořena.

Před samotným spuštěním se naplní kolekce dat, kde podle četností jsou duplikovány jednotlivé hodnoty generovaných dat, a následně je vybírána podle náhodně vygenerovaného indexu náhodná hodnota. Hodnoty s větší četností se opakují častěji, proto jsou zpravidla vybrány. Po této inicializaci se spustí již samotné generování.



Obrázek 8: Generování klienta. Zdroj: autor.

Nejprve je náhodně vybráno křestní jméno, k tomuto křestnímu jménu se dohledá pohlaví a následně podle pohlaví se již hledá náhodné příjmení, ale již omezené podmínkou pohlaví. Ke křestnímu jménu a příjmení se náhodně vybere rok narození. Dále se generuje datum. Ten je již generován zcela náhodně, není ovlivněn četnostmi jako rok narození. Vygenerované datum avšak musí existovat v kalendářním roce, tj. nemůže se stát, že by se například vygeneroval 13. měsíc apod. Podle údajů data narození a pohlaví se vygeneruje rodné číslo, které splňuje dělitelnost 11 a odpovídá konvenci příslušného pohlaví. V dalším kroku se podle četností vygeneruje město. Logicky nejčastěji generovaným městem je Praha. K tomuto městu se náhodně dohledá ulice a číslo popisné, toto dohledávání je také již plně náhodné a není závislé na četnostech. K takto vygenerovanému klientovi se již pouze doplní mobilní číslo, které také splňuje konvence českých mobilních čísel. Takto vygenerovaný uživatel je následně vložen do nově vytvořené tabulky.

Spuštění modulu

```
--vygeneruje 100 náhodných uživatelů do nově vytvořené tabulky
user list_gen
BEGIN
 package_random_table.generate(100,'user_list_gen');
commit;
end;
```

Kód 2: Generování náhodných klientů

Ukázka výstupních dat:

Jméno	Příjmení	RČ	Datum narození	Město	Ulice	Č.P.	Mobil
TOMÁŠ	DUONG	6808312236	31.8.1968	TEPLICE	NA VÝŠINÁCH	1379	603916032
EVA	ŠTĚPÁNKOVÁ	3458070803	7.8.1934	OPATOVICE	NOVÁ	98	721862935
VIKTORIE	VOLDŘICHOVÁ	5251162532	16.1.1952	KROMĚŘÍŽ	RAISOVA	2213	736907031
MIROSLAV	ŠNAJDR	7811111011	11.11.1978	PRAHA	NA DOLÍKU	14	608947618

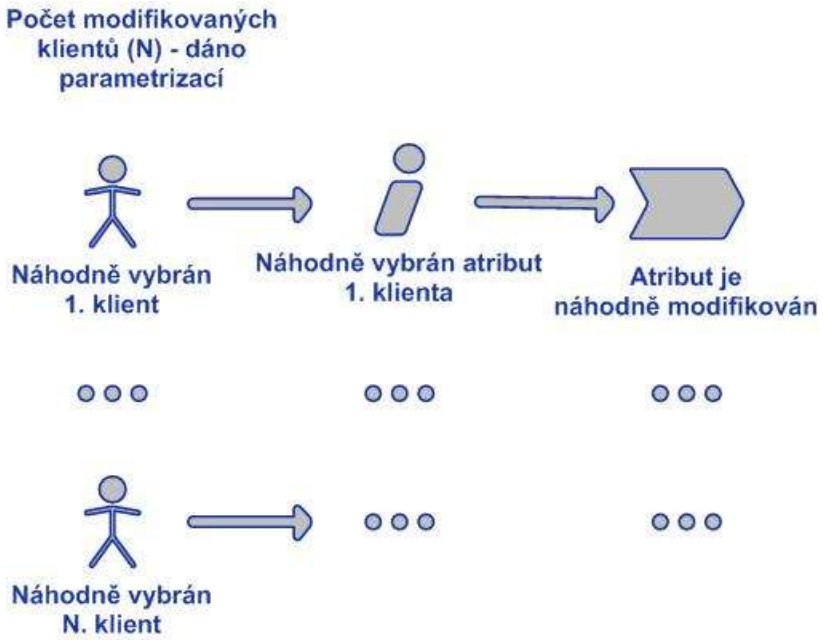
Tabulka 25: Ukázka vygenerovaných náhodných klientských dat generátorem. Zdroj: autor.

4.1.2 Upravující modul (Dirty Module)

Tento modul náhodně upravuje vygenerovaná klientská data. Úpravy jsou náhodného charakteru a výsledkem by mělo být přiblížení se k reálným datům, která se vyskytují v reálných klientských tabulkách. Úpravy jsou charakteru překlepů, špatně vyplněných jednotlivých údajů apod. Podle parametrizace lze nastavit, jak velké procento záznamů bude poupraveno.

4.1.2.1 *Postup úpravy dat*

Nejprve se načte z parametrizace údaj, kolik klientů má být upraveno. Následně se provede tento počet úprav v tabulce user_list, kde jsou uloženi klienti, kteří jsou předmětem úpravy. Pro každou úpravu je náhodně vybráno jedno z dirty (upravujících) pravidel. Každé pravidlo má také rozdílnou pravděpodobnost provedení. Tato pravidla jsou uložena v tabulce tab_dirty_rules a lze je dynamicky rozšiřovat nebo měnit jejich pravděpodobnosti provedení.



Tabulka 26: Postup náhodné úpravy dat, tak aby odpovídaly reálným. Zdroj: autor.

Takto je vyplněná parametrizace pravidel:

Název pravidla	Váha pravidla pro DIRTY MODUL	Váha pravidla pro duplikační MODUL	Pro textovou hodnotu (Y/N)	Pro datumovou hodnotu (Y/N)	Pro číselnou hodnotu (Y/N)
INSERT_CHAR	5		Y	N	N
REPLACE_ CHAR	5		Y	N	Y
REPLACE_ VALUE_S	2	2	Y	Y	Y
REPLACE_ VALUE_D	1		Y	N	N
DELETE	1	1	Y	Y	Y

Tabulka 27: Dirty (upravujících) pravidel používaných v rámci Dirty modulu. Zdroj: autor.

Sloupec váha určuje, jak často se bude dané pravidlo používat: čím vyšší je hodnota, tím častěji bude dané pravidlo použito. Každé z pravidel podle své povahy je určeno pro úpravu určitého datového typu, proto jsou zde sloupce, podle kterých je možno rozpoznat,

jaký typ atributu klienta může být tímto pravidlem změněn. Z tabulky je zřejmé, že je také používán pro duplikační modul a sloupce.

Jednotlivá pravidla:

a) INSERT_CHAR – pravidlo, které náhodně vkládá znak do textové hodnoty, je určeno pouze pro textové řetězce.
b) REPLACE_CHAR – pravidlo, které náhodně nahrazuje znak v textové hodnotě, je určeno pro textové řetězce i číselné řetězce.
c) REPLACE_VALUE_S – pravidlo, které nahrazuje celou hodnotu hodnotou ze stejného číselníku, ze kterého pochází i původní hodnota, určeno pro jakýkoliv typ hodnoty.
d) REPLACE_VALUE_D – pravidlo, které nahrazuje celou hodnotu hodnotou z náhodně vybraného číselníku, určeno pro textové řetězce.
e) DELETE – pravidlo, které danou hodnotu jednoduše smaže, lze provést pro jakýkoliv typ hodnoty.

Spuštění modulu

```
BEGIN
      pck_make_dirty.main;
      COMMIT;
END;
```

Kód 3: Spuštění upravujícího modulu (Dirty module)

Klientská data před spuštěním vychází z minulého příkladu klientských dat.

Klientská data po spuštění:

Jméno	Příjmení	RČ	Datum narození	Město	Ulice	Č.P.	Mobil
TOMÁŠ	DUANG	6808312236	31.8.1968	TEPLICE	NA VÝŠINÁCH	1379	603916032
KUNÍN	NOVÁK	3458070803	7.8.1934	OPATOVICE	NOVÁ	98	721862935
VIKTORIE	VOLDŘICHOVÁ	5251162532	13.2.1970	KROMĚŘÍŽ	RAISOVA	2213	736907031
MIROSLAV	ŠNAJDR	7811111011	11.11.1978		NA DOLÍKU	14	608947618

Tabulka 28: Klientská data po spuštění Dirty modulu. Zdroj: autor.

Pro větší přehlednost jsou změněné údaje označené světlým odstínem.

Část záznamů z logující tabulky tab_changed_item je následující:

Id klienta	Název sloupce	Původní hodnota	Nová hodnota	Text
1	SURNAME	DUONG	DUANG	Dirty rule: REPLACE_CHAR
2	FIRST_NAME	EVA	KUNÍN	Dirty rule: REPLACE_VALUE_D

2	SURNAME	ŠTĚPÁNKOVÁ	NOVÁK	Dirty rule: REPLACE_VALUE_S
3	BIRTH_DATE	16.1.1952	13.2.1970	Dirty rule: REPLACE_VALUE_S
4	TOWN	PRAHA		DELETE

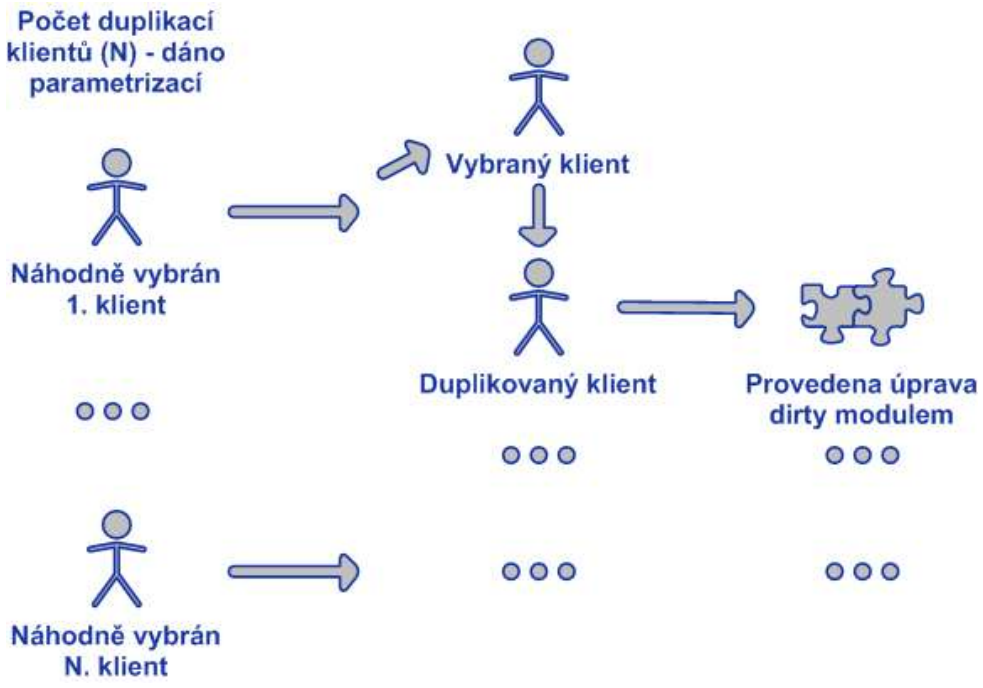
Tabulka 29: Záznamy z logující tabulky o proběhlých změnách provedených Dirty modulem. Zdroj: autor.

4.1.3 Duplikační modul (Duplicate Module)

Tento modul na upravených datech vytváří navíc jejich duplikáty. V rámci procesu duplikování samozřejmě provádí ještě další transformace. Například u duplikovaných klientů vygeneruje novou adresu nebo pro ženy změní příjmení a adresu. Jedná se o další stupeň úpravy tak, aby se opět více blížila reálným datům. I tato pravidla jsou parametrizovatelná a lze jim přiřazovat pravděpodobnost provedení.

4.1.3.1 Postup duplikace klientských záznamů

Nejprve se určí počet záznamů na základě parametrizace, které mají být duplikovány, a následně se provádí samotná duplikace. Náhodně se určí klient, pro kterého se vytvoří kopie a nad touto kopií se provede nějaká z úprav. Tyto úpravy vychází ze společné parametrizace s upravujícím modulem, ale pravděpodobnosti zvolení modifikačního pravidla jsou vlastní. Jednotlivé provedené úpravy se opět zaznamenají do pomocné tabulky tab_changed_item, odkud lze dohledat, který záznam a jakým pravidlem byl změněn.



Obrázek 9: Duplikace náhodně vybraného klienta. Zdroj: autor.

Parametrizace:

Název pravidla	Váha pravidla pro DIRTY MODUL	Váha pravidla pro duplikační MODUL	Poznámka
REPLACE_VALUE_S	2	2	Pravidlo použité pro DIRTY i duplikační modul
DELETE	1	1	Pravidlo použité pro DIRTY i duplikační modul
SAME		3	
DIFF_ADR		5	
DIFF_SUR_Z		6	
SAME_ADDR_ DIFF_OTH		2	

Tabulka 30: Sdílená parametrizace změnových pravidel používaná při duplikaci. Zdroj: autor.

Sloupec váha pro duplikační modul má stejný význam jako při používání upravujícího modulu, jen jsou parametrizace oddělené, aby bylo možno přiřadit různé pravděpodobnosti.

Jednotlivá pravidla:

a) REPLACE_VALUE_S – nahrazení celé jedné hodnoty atributu klienta hodnotou jinou stejného typu.
b) DELETE – smazání jedné hodnoty atributu klienta.
c) SAME – dojde k pouhé duplikaci záznamu a již se neprovede žádná změna.
d) DIFF_ADR – u kopie klienta se vygeneruje nová celá platná adresa.
e) DIFF_SUR_Z – u klientky se změní příjmení na jiné platné ženské příjmení.
f) SAME_ADDR_DIFF_OTH – u kopie klienta se ponechá pouze adresa a všechny ostatní údaje se přegenerují, bude se tedy jednat o nového klienta, který ale přebývá na stejné adrese.

Parametrizace lze samozřejmě rozšířit a doplnit další pravidla, aby data byla ještě reálnější.

Spuštění modulu

```
BEGIN
      pck_make_dup.main;
      COMMIT;
END;
```

Kód 4: Spuštění duplikačního modulu (Duplicate module)

Klientská data před spuštěním vychází z minulého příkladu klientských dat.

Klientská data po spuštění:

Jméno	Příjmení	RČ	Datum narození	Město	Ulice	Č.P.	Mobil
TOMÁŠ	DUANG	6808312236	31.8.1968	Teplice	NA VÝŠINÁCH	1379	603916032
KUNÍN	NOVÁK	3458070803	7.8.1934	Opatovice	NOVÁ	98	721862935
VIKTORIE	VOLDŘICHOVÁ	5251162532	13.2.1970	Kroměříž	RAISOVA	2213	736907031
MIROSLAV	ŠNAJDR	7811111011	11.11.1978		NA DOLÍKU	14	608947618
VIKTORIE	ŽERNÍČKOVÁ	5251162532	13.2.1970	Kroměříž	RAISOVA	2213	736907031
MIROSLAV	ŠNAJDR	7811111011	11.11.1978		NA DOLÍKU	14	608947618

Tabulka 31: Klientská data po spuštění Deduplikačního modulu. Zdroj: autor.

Pro větší přehlednost jsou změněné údaje označené světlým odstínem.

Část záznamů z logovací tabulky tab_changed_item je následující:

Id klienta	Název sloupce	Původní hodnota	Nová hodnota	Text	Id kopie klienta
3				Duplication	5
5	SURNAME	VOLDŘICHOVÁ	ŽERNÍČKOVÁ	Dirty rule: DIFF_SUR_Z	
4				Duplication	6
6				Dirty rule: SAME	

Tabulka 32: Záznamy z logovací tabulky o proběhlých změnách provedených Duplikačním modulem. Zdroj: autor.

4.1.4 Čistící modul (Clean Module)

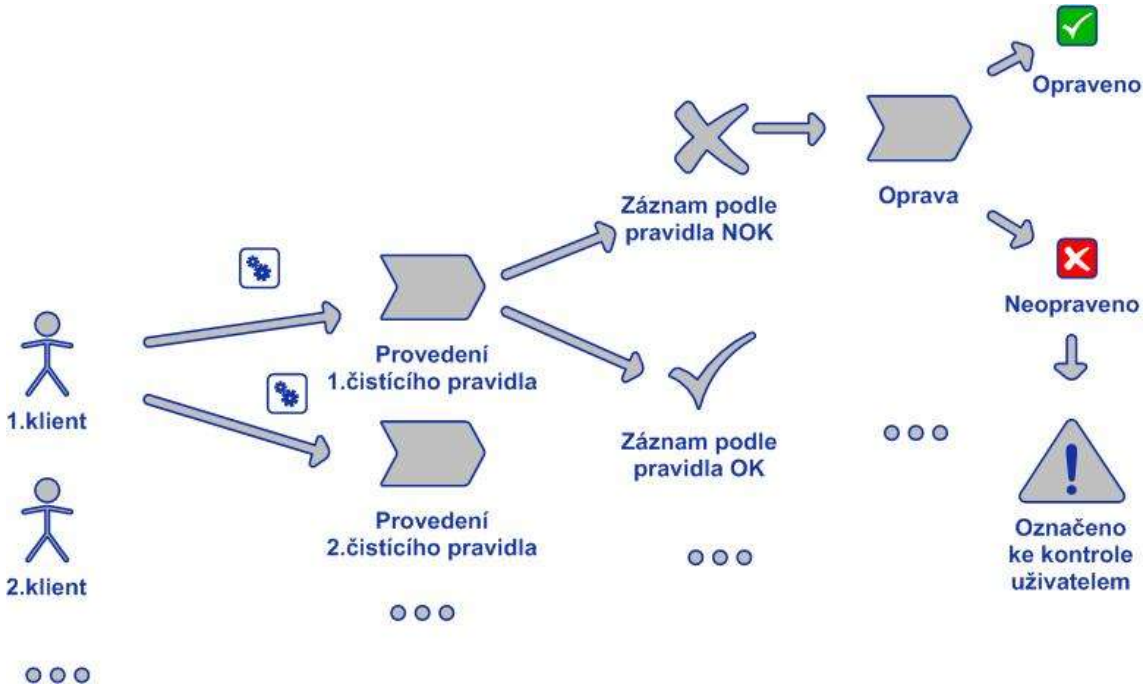
Tento modul patří již do druhé části skriptu, která slouží ke zkvalitňování dat. Zde probíhají různé kontroly a v případě nalezení nesrovnalostí se snaží data opravit nebo přinejmenším je označit pro uživatele ke kontrole s možnými návrhy oprav. Tento modul navíc snižuje záznamu po opravě koeficient čistoty, aby bylo zřejmé, z jak velké části byl záznam opraven. Nízký koeficient čistoty udává, že záznam nebylo možné opravit a je ve velmi špatné kvalitě. Čistící modul připravuje data pro deduplikační modul, aby data před samotnou deduplikací byla bez zjevných chyb.

4.1.4.1 Postup kontroly a čištění klientských záznamů

Jednotlivé klientské záznamy jsou postupně procházeny a na každý jsou aplikovány postupně všechna čistící pravidla. Jednotlivá čistící pravidla se skládají ze dvou kroků:

- Prvním krokem je kontrola hodnot daného klienta. Jednotlivé porovnávané hodnoty klienta se zpracovávají a nakonec se vzájemně porovnávají. V případě neshody přichází na řadu druhý krok čištění. V případě shody se přechází k dalšímu čistícímu pravidlu.
- Druhým krokem je pokus o opravu chybné hodnoty klienta. Pokud se oprava provede, zaznamená se k danému klientovi rating úspěšně opraveného záznamu, pokud se oprava nepodaří, zaznamená se k danému klientovi rating neopraveného

záznamu. Koeficient čistoty se mění z toho důvodu, aby bylo na první pohled zřejmé, že záznam buď není v pořádku, nebo byl skriptem upraven.



Obrázek 10: Postup čištění jednotlivých klientů Clean modulem. Zdroj: autor.

Názornější je konkrétní ukázka parametrizace čistících pravidel:

Identifikátor pravidla	Procedura pro opravení chybné hodnoty	Krátký popis
CHECK_IN_LIST_FNAME	REP_CHECK_IN_LIST	Pravidlo kontrolující křestní jméno (first_name) v číselníku.
CHECK_IN_LIST_SURNAME	REP_CHECK_IN_LIST	Pravidlo kontrolující příjmení (surname) v číselníku.
CHECK_IN_LIST_TOWN	REP_CHECK_IN_LIST	Pravidlo kontrolující město v číselníku.
BIRTH_RULE	REP_BIRTH	Pravidlo porovnávající datum narození a rodné číslo.
CHECK_GENDER	REP_GENDER	Pravidlo kontrolující návaznost rodného čísla, jména a příjmení.
CHECK_ADDRESS	REP_ADDR	Pravidlo kontrolující smysluplnost celé adresy.
CHECK_MOB	REP_MOB	Pravidlo kontrolující formát mobilního čísla.

Tabulka 33: Ukázka z parametrizace čistících pravidel. Zdroj: autor.

Popis pravidel:

CHECK_IN_LIST_FNAME, CHECK_IN_LIST_SURNAME, CHECK_IN_LIST_TOWN

Toto pravidlo kontroluje, zda se dané křestní jméno/příjmení/město vyskytuje v příslušném číselníku. Pravidlo se skládá ze dvou částí, a aby bylo pravidlo splněno, musí se obě části pravidla shodovat.

Části:

i. Část vybírající hodnotu ze záznamu daného klienta
ii. Část dohledávající hodnotu klienta v číselníku

Po vyhodnocení neshody je volána opravná funkce REP_CHECK_IN_LIST, která zkouší vyhledat v číselníku příslušného typu (v tomto případě v číselníku křestních jmen) co nejpodobnější záznamy. Tato opravná funkce je pro všechny tři čistící pravidla shodná. Právě determinace shody jednotlivých záznamů je velmi náročná operace a je třeba prohledávat velké množství dat. Pro porovnávání jednotlivých textových hodnot je možno použít různé srovnávací algoritmy, například JARO WINKLER funkce podobnosti, Levenshteinova vzdálenost a další. V určování podobnosti řetězců v opravné funkci byl použit algoritmus pro učení Levenshteinovy vzdálenosti. (Rahm, 2000)

V opravné funkci je logika, která na základě hodnot Levenshteinovy vzdálenosti nalezne nejpodobnější hodnotu. Pokud je odlišnost malá a kandidát na shodu je jeden, pak je nalezená hodnota automaticky nahrazena. Pokud je odlišnost kandidátů od hodnoty klienta velká, respektive klientů je velké množství, pak je tento záznam označen ke kontrole uživatelem.

BIRTH_RULE

Toto pravidlo kontroluje vazbu mezi rodným číslem a datem narození. Opět vrácené hodnoty jednotlivých částí pravidel se musí vzájemně shodovat, aby pravidlo bylo splněno. Části pravidla:

i. Část převádějící datum narození na definovaný textový řetězec
ii. Část generující z rodného čísla datum v definované textové podobě
iii. Část kontrolující dělitelnost rodného čísla 11 a vracející opět datum v definovaném textovém řetězci

Po vyhodnocení neshody je volána opravná funkce REP_BIRTH. Tato funkce se pokusí datum narození opravit z rodného čísla, které musí být ovšem v pořádku, tj. musí splňovat dělitelnost jedenácti. Pokud ani tato podmínka není splněna, je záznam označen za neopravitelný a je předán uživateli k revizi. Pokud je splněna, pak je datum narození opraveno podle rodného čísla.

CHECK_GENDER

Toto pravidlo porovnává pohlaví, které je uloženo v rodném čísle, příjmení a jméně. Typ pohlaví z těchto atributů vytáhne a následně je porovná. Oprava v případě neshod je netriviální, proto případná chyba je předána uživateli ke kontrole.

CHECK_ADDRESS

V tomto pravidle je kontrolováno, zda v daném městě je daná ulice a zda v dané ulici existuje dané číslo popisné. V případě jakékoliv neshody je snaha dohledat co nejpodobnější název ulice v daném městě, popřípadě dohledat co nejbližší číslo popisné v dané ulici. Pokud je návrhů více a pravidlo není schopno automaticky vybrat nejlepšího kandidáta, je výčet možností předán uživateli k rozhodnutí.

CHECK_MOB

Je kontrolován formát mobilního telefonního čísla. Pokud číslo formát nesplňuje, je označeno za chybné. Bohužel není možné mobilní číslo opravit, protože neexistuje vzor.

Parametrizace čistících pravidel je možné dále upravovat a rozšiřovat.

Spuštění modulu

```
BEGIN
      pck_make_clean.main;
      COMMIT;
END;
```

Kód 5: Spuštění čistícího modulu (Clean Module)

Klientská data před spuštěním vychází z minulého příkladu klientských dat.

Klientská data po spuštění:

Jméno	Příjmení	RČ	Datum narození	Město	Ulice	Č.P.	Mobil
TOMÁŠ	DUANG	6808312236	31.8.1968	Teplice	NA VÝŠINÁCH	1379	603916032
KUNÍN	NOVÁK	3458070803	7.8.1934	Opatovice	NOVÁ	98	721862935
VIKTORIE	VOLDŘICHOVÁ	5251162532	16.01.1952	Kroměříž	RAISOVA	2213	736907031
MIROSLAV	ŠNAJDR	7811111011	11.11.1978		NA DOLÍKU	14	608947618
VIKTORIE	ŽERNÍČKOVÁ	5251162532	16.01.1952	Kroměříž	RAISOVA	2213	736907031
MIROSLAV	ŠNAJDR	7811111011	11.11.1978		NA DOLÍKU	14	608947618

Tabulka 34: Klientská data po spuštění Clean (čistícího) modulu. Zdroj: autor.

Pro větší přehlednost jsou změněné údaje označené světlým odstínem.

Část záznamů z logovací tabulky tab_changed_item je následující:

Id klienta	Název sloupce	Původní hodnota	Nová hodnota	Text
1	FIRST_NAME	KUNÍN	CHOOSE:ERVÍN	CHECK_IN_LIST_FNAME
3	BIRTH_DATE	13.2.1970	16.01.1952	BIRTH_RULE

Tabulka 35: Záznamy z logovací tabulky o proběhlých změnách provedených Clean modulem. Zdroj: autor.

Ze záznamů čistícího modulu je vidět, že datum narození bylo správně opraveno podle rodného čísla. Avšak druhý chybný záznam „Kunín“ již opraven nebyl. Jedná se o hodnotu, která vznikla úplným přepsáním křestního jména hodnotou města, proto je velmi náročné najít původní správnou hodnotu. Čistící modul se alespoň snaží dát návrhy k opravě, což je vidět z tabulky. Modul navrhuje opravit hodnotu „Kunín“ křestním jménem „Ervín“.

Samozřejmě při rozpoznávání chyb mohou vznikat i špatné opravy nebo návrhy na špatné opravy. Viz následující tabulka:

Id klienta	Název sloupce	Původní hodnota	Nová hodnota	Text	Modul
15	SURNAME	KINDLOVÁ	KINXDULOVÁ	INSERT_CHAR	DIRTY
18	SURNAME	KINXDULOVÁ	CHOOSE:KORDULOVÁ, RANDULOVÁ	CHECK_IN_LIST_ SURNAME	CLEAN

Tabulka 36: Záznamy z logovací tabulky o změnách provedených programem. Zdroj: autor.

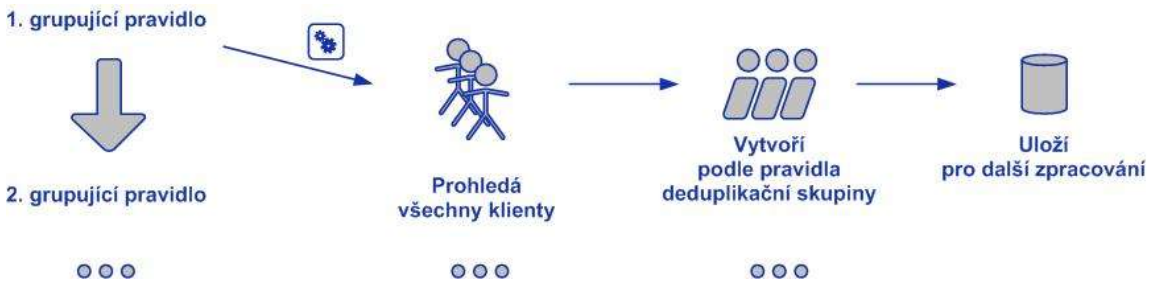
Tento případ je ukázkou toho, jak se čistící modul zachová při nalezení více kandidátů na opravu. Modul do sloupce „Nová hodnota“ vloží klíčové slovo choose (v překladu vyber) a nejlepší kandidáty, tj. návrhy „Kordulová“ a „Randulová“. Tato změna (znehodnocení příjmení) byla provedena právě takovým způsobem, že návrhy na opravu již neobsahují původní hodnotu. Je otázka, zda by i člověk byl tuto změnu schopen rozpoznat.

4.1.5 Deduplikační modul (Deduplicate Module)

Tento modul následuje po čistícím modulu a dohledává v datech duplicitní klienty. Musí se srovnávat podobnosti záznamů podle stanovených pravidel a vyhodnocovat je. Následně jsou tyto podobnosti tříděny podle hodnotících funkcí tak, aby byl nalezen nejpodobnější záznam. Pokud některý z kandidátů podobnosti překročí stanovenou mez podobnosti, je automaticky považován za kopii daného klienta.

4.1.5.1 *Postup rozpoznání duplicitního záznamu – deduplikace klientů*

Identifikace deduplikačních skupin



Obrázek 11: Vytvoření deduplikačních skupin. Zdroj: autor.

Po spuštění deduplikačního skriptu se nejprve vytvoří skupiny podobných klientů podle různých grupujících pravidel, tj. postupně se prochází jednotlivá pravidla pro vytváření skupin a podle každého pravidla se vytvoří skupiny klientů takové, které mají více než jednoho člena. To znamená, že do skupiny musí být přiděleni alespoň dva klienti. Ukázka z parametrizace pravidel pro vytváření deduplikačních skupin:

Id grupovacího pravidla	Sloupce, podle kterých se vytvářejí skupiny	Rating	Popis
FI_BIRTHN_DEDU	tab.first_name, tab.birth_number	3	Deduplikační pravidlo podle jména a rodného čísla
BIRTHD_BIRTHN_DEDU	tab.birth_date, tab.birth_number	1	Deduplikační pravidlo podle rodného čísla a datumu narození
TO_ST_HN_DEDU	tab.town,tab.street, tab.house_number	2	Deduplikační pravidlo podle bydliště, tj. města, ulice a č.p.
MOB_DEDU	tab.mobile_number	1	Deduplikační pravidlo podle mobilního čísla
SU_BIRTHN_DEDU	tab.surname, tab.birth_number	2	Deduplikační pravidlo podle příjmení a rodného čísla
FI_SU_DEDU	tab.surname, tab.first_name	2	Deduplikační pravidlo podle příjmení a křestního jména

Tabulka 37: Parametrizace pro vytváření deduplikačních skupin. Zdroj: autor.

Jednotlivé deduplikační skupiny jsou tvořeny jedním nebo více atributy klientů. Tedy pokud dva klienti mají tyto atributy totožné, vytvoří se podle daného deduplikačního pravidla pro ně skupina s jednoznačným id.

Ukázka vytvoření deduplikačních skupin

Klienti:

Id klienta	Jméno	Příjmení	RČ	
22	MILUŠE	WETTEROVÁ	5759243457	…
23	ZDENĚK	KNEIFEL	6504021007	…
24	KARINA	VONDŘEJCOVÁ	7160052504	…
25	KARINA	VONDŘEJCOVÁ	7160052504	…

Tabulka 38: Vzoroví klienti pro ukázání vzniku deduplikačních skupin. Zdroj: autor.

Deduplikační skupiny:

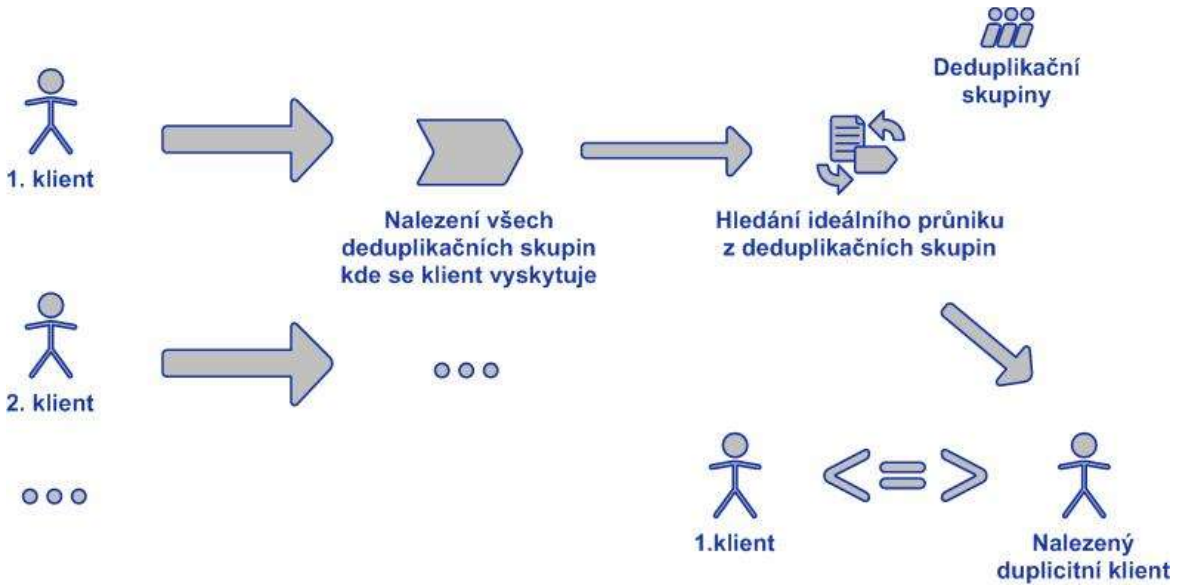
Id klienta	Id vzniklé skupiny	Id grupovacího pravidla	Id klientů vyskytující se ve skupině	Rating pravidla	Počet klientů ve skupině
24	10	FI_SU_DEDU	24,25	2	2
25	10	FI_SU_DEDU	24,25	2	2

Tabulka 39: Ukázka vzniklých deduplikačních skupin podle pravidla FI_SU_DEDU. Zdroj: autor.

Pro klienta s id 24 a s id 25 se vytvořila jedna deduplikační skupina. Tato skupina má id 10 a případně bude dále použita pro slučovaní jednotlivých deduplikačních skupin a rozhodování zda tito dva klienti jsou totožní.

Pro každé deduplikační pravidlo (viz tabulka deduplikačních pravidel) jsou podle tohoto pravidla vytvořeny skupiny klientů. Tyto skupiny klientů se shodují v atributech definovaných v deduplikujícím pravidlu. Jednotlivé skupiny klientů jsou jednoznačně identifikovatelné přes id skupiny a id grupovacího pravidla. Toto byl první krok deduplikačního modulu.

Identifikace duplicitních klientů

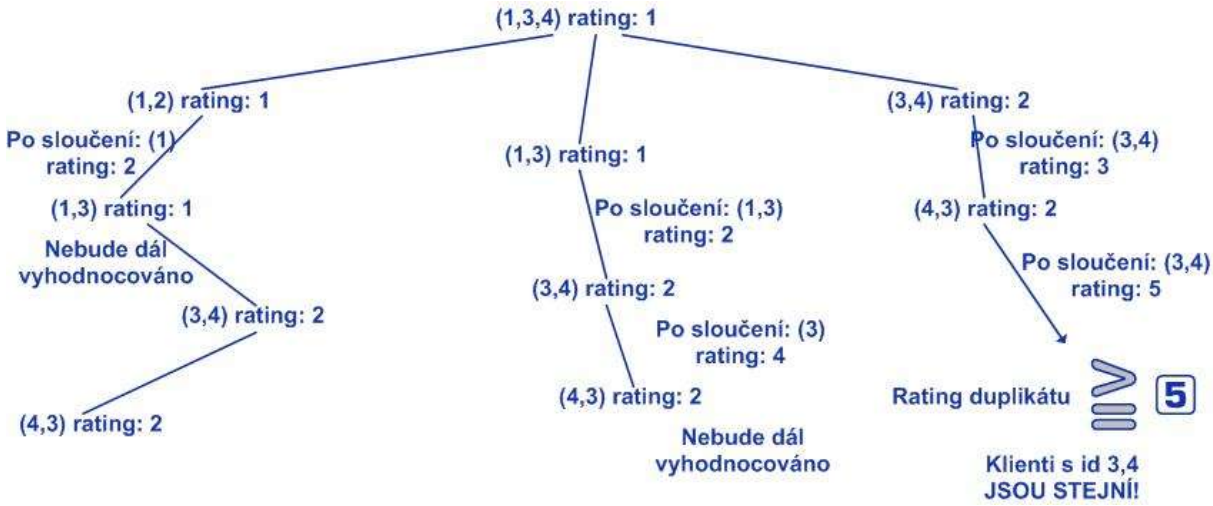


Obrázek 12: Algoritmus rozpoznání duplicitního klienta. Zdroj: autor.

Druhým krokem je postupné slučování přes jednotlivé vzniklé deduplikační skupiny jednoho klienta a postupně jsou pro tyto skupiny vytvářeny průniky. Skupiny se procházejí v pořadí od nejvíce ohodnocených ratingem po nejméně ohodnocené. Rating odpovídá váze jednotlivého pravidla. Při sloučení dvou různých deduplikačních skupin se následně ratingy sčítají.

Postupné slučování klientů odpovídá procházení grafu, kde na následujícím uzlu je proveden průnik s další deduplikační skupinou. Pokud již neexistuje žádná další skupina ke sloučení, je vyhodnocen rating doposud sloučených skupin a je porovnán s minimálním ratingem, který je potřeba dosáhnout, aby se jednalo o shodné klienty. Pokud je rating sloučených skupin větší než prahový rating a samozřejmě pokud je ve skupině sloučených skupin více než jeden klient, pak tento klient/tito klienti jsou vyhodnoceni jako shodní klienti. Pokud rating sloučených skupin není větší než prahový rating, snaží se algoritmus vytvořit jiný průnik klientů tak, aby tento minimální rating byl dosažen. Jinou variantou špatné cesty v grafu je, když po slučování zůstal v uzlu pouze jediný hledaný klient, pak se musíme také snažit najít jinou kombinaci slučování tak, aby konečná množina obsahovala alespoň dva klienty. Pokud se to v žádném případě nepodaří, pak pravděpodobně neexistuje podobný klient ke klientu hledanému.

Ukázka postupu slučování na jednoduchém grafu.



Obrázek 13: Postup postupného slučování rozpoznaných duplikačních skupin. Zdroj: autor.

První cesta v grafu slučuje skupiny kontaktů (1,3,4) a (1,2). Výsledkem tohoto sloučení je samotný kontakt 1. Druhá cesta se nejprve snaží sloučit skupiny kontaktů (1,3,4) a (1,3). Výsledkem tohoto sloučení je skupina (1,3). Následně je s tímto výsledkem sloučena skupina (3,4), pak je výsledkem sloučení samotný kontakt (3). Třetí cestou je postupné sloučení skupin (1,3,4), (3,4) a (4,3). Výsledkem této cesty je skupina kontaktů (3,4), tato cesta má výsledný rating 5. Tento rating dosáhl prahové hodnoty 5 a klienti s id 3 a 4 jsou považováni za duplikáty.

Tento druhý krok deduplikačního modulu (provádění identifikace podobných klientů) je prováděno pro všechny klienty v tabulce user_list, pro které vznikly deduplikační skupiny.

Spuštění modulu

```
BEGIN
      pck_make_dedup.main;
      COMMIT;
END;
```

Kód 6: Spuštění deduplikačního modulu (Deduplicate module)

Klientská data před spuštěním vychází z minulého příkladu klientských dat.

Níže jsou výpisy z tabulky tab_ded_group, jedná se o tabulku deduplikačních skupin a je v ní vidět rozpoznané skupiny. Je zřejmé, že u kontaktu s id 4 a 6 se jedná o naprosto totožné klienty, protože byly vytvořeny deduplikační skupiny podle všech pravidel. U kontaktů 3 a 5 je zřejmé, že nebyla vytvořena deduplikační skupina podle příjmení, protože to je různé. Pravděpodobně se jedná o ženu, která se provdala, ale všechny ostatní informace o ní zůstaly nezměněné.

Id skupiny	Deduplikační pravidlo	Kontakty ve skupině	Rating skupiny
3	MOB_DEDU	3,5	1
3	BIRTHD_BIRTHN_DEDU	3,5	1
3	TO_ST_HN_DEDU	3,5	2
3	FI_BIRTHN_DEDU	3,5	3
		SUMA ratingů	**7**

Tabulka 40: Ukázka vzniklých deduplikačních skupin pro klienty s id 3 a 5. Zdroj: autor.

Id skupiny	Deduplikační pravidlo	Kontakty ve skupině	Rating skupiny
4	BIRTHD_BIRTHN_DEDU	4,6	1
4	TO_ST_HN_DEDU	6,4	2
4	MOB_DEDU	6,4	1
4	FI_BIRTHN_DEDU	6,4	3
5	FI_SU_DEDU	6,4	2
		SUMA ratingů	**9**

Tabulka 41: Ukázka vzniklých deduplikačních skupin pro klienty s id 4 a 6. Zdroj: autor.

Klientská data po spuštění:

Jméno	Příjmení	RČ	Datum narození	Město	Ulice	Č.P.	Mobil
TOMÁŠ	DUANG	6808312236	31.8.1968	Teplice	NA VÝŠINÁCH	1379	603916032
KUNÍN	NOVÁK	3458070803	7.8.1934	Opatovice	NOVÁ	98	721862935
VIKTORIE	VOLDŘICHOVÁ	5251162532	16.01.1952	Kroměříž	RAISOVA	2213	736907031
MIROSLAV	ŠNAJDR	7811111011	11.11.1978		NA DOLÍKU	14	608947618
VIKTORIE	ŽERNÍČKOVÁ	5251162532	16.01.1952	Kroměříž	RAISOVA	2213	736907031
MIROSLAV	ŠNAJDR	7811111011	11.11.1978		NA DOLÍKU	14	608947618

Tabulka 42: Klientská data po spuštění Deduplikačního modulu. Zdroj: autor.

Pro větší přehlednost jsou změněné údaje označené světlým odstínem.

Část záznamů z logovací tabulky tab_changed_item je následující:

Id klienta	Název sloupce	Původní hodnota	Nová hodnota	Text	Id kopie klienta
3				Deduplication	5
4				Deduplication	6

Tabulka 43: Záznamy z logovací tabulky o proběhlých změnách provedených Deduplikačním modulem. Zdroj: autor.

4.2 Podpůrné služby

4.2.1 Záznamy změn v tabulce klientů

Části podpůrných služeb byly už popsány. Jednalo se o detailní zaznamenávání změn nad tabulkou user_list. Záznamy o těchto změnách jsou velmi užitečné, protože je možné postupně sestavit historii změn daného kontaktu a tak zjistit, zda čistící a deduplikační modul nalezly správné hodnoty. Tyto změny jsou zaznamenávány do tabulky TAB_CHANGED_ITEM.

Toto by u reálných dat nebylo možné, protože ve velkém množství dat není možné dohledat chyby. Pouze bychom se mohli dohadovat, zda opravdu všechny chyby byly nalezeny.

4.2.2 Záznamy běhů jednotlivých modulů

Pomocná funkčnost je zapouzdřena v balíčku PCK_RUN_LOG, kde se zaznamenává začátek běhu jednotlivých modulů, jejich průběh, případně vzniklé chyby… Nakonec je zaznamenán i čas skončení běhu modulu. Je zde také zaznamenáno, kolik záznamů klientů v rámci jednoho běhu bylo zpracováno. Všechny tyto záznamy se ukládají do tabulky TAB_RUN_LOG a případné podrobné události do tabulky TAB_RUN_LOG_DETAIL.

4.2.3 Debugovací funkce

Poslední funkčností podpůrných služeb je debugovací funkce, která slouží k ladění a detailnímu zaznamenávání běhů jednotlivých modulů. Je zde možno dohledat jednotlivé kroky skriptů, a tak najít případnou chybu. Tato funkce je velice užitečná, protože vzhledem k náhodnosti prováděných příkazů není možné po doběhnutí modulu znovu zopakovat proběhlý scénář.

4.3 Postup nasazení modulů

4.3.1 Nasazení generátoru náhodných dat

Na přiloženém CD jsou ve složce s názvem D:\prakticka_cast\generator_dat uloženy všechny zdrojové kódy pro náhodný generátor dat. Pro nasazení generátoru dat stačí spustit v příkazové řádce aplikace SQLPLUS skript s názvem: LOAD_EVERYTHING.SQL

Ukázka:

SQL> @D:\prakticka_cast\generator_dat\LOAD_EVERYTHING.sql

Kód 7: Spuštění nasazení generátoru náhodných klientů v SQL*Plus

4.3.2 Nasazení všech ostatních modulů

Na přiloženém CD jsou ve složce s názvem D:\prakticka_cast\duplikator_deduplikator uloženy všechny zdrojové kódy pro dirty modul, duplikační modul, clean modul i deduplikační modul. Pro nasazení všech těchto modulů stačí spustit v příkazové řádce aplikace SQLPLUS skript s názvem: LOAD_EVERYTHING.SQL

Ukázka:

SQL> @D:\prakticka_cast\duplikator_deduplikator\LOAD_EVERYTHING.sql

Kód 8: Spuštění nasazení všech modulů v SQL*Plus

Ukázkové spuštění všech modulů

Všechny moduly lze najednou spustit přes jednotlivá volání jejich funkčností. Proběhne tedy nejprve vygenerování 100 náhodných klientů, ti jsou následně upraveni a zduplikováni. Následuje pak proces čištění a proces deduplikace. Základní informace o proběhlých procesech jsou uloženy v tabulce TAB_RUN_LOG

```
BEGIN
      --vygenerování 100 náhodných klientů do tabulky user_list
      package_random_table.generate(100,'user_list');
```

```
        pck_make_dirty.main;
        pck_make_dup.main;
        pck_make_clean.main;
        pck_make_dedup.main;
        COMMIT;
END;
```

Kód 9: Komplexní spuštění všech skriptů praktické části

Ukázkové záznamy o běžích modulech:

Id záznamu	Module	Text	Čas začátku	Čas konce	Počet zpracovaných záznamů	Počet chyb
1	DIRTY	Module has STOPED	09.09.16 19:43:13	09.09.16 19:43:13	10	0
2	DUP	Module has STOPED	09.09.16 19:43:13	09.09.16 19:43:15	10	0
3	CLEAN	Module has STOPED	09.09.16 19:43:15	09.09.16 19:43:50	19	0
4	DEDUP	Module is running	09.09.16 19:43:50	09.09.16 19:43:51	10	0

Tabulka 44: Záznamy o běžích jednotlivých modulů. Zdroj: autor.

4.4 Vlastní postup měření

V této kapitole bude popsáno, jak byl proveden test posouzení zkvalitnění klientských dat po běhu čistících skriptů. Pro toto posouzení budou použity některé míry z ISO normy SQUARE.

4.4.1 Příprava hodnocení

Účelem hodnocení je dokázat, že čistící skripty plní svůj účel, tj. data čistí a deduplikují. V rámci vyhodnocení bude zkoumáno, jakou míru znehodnocení a duplikací je schopen čistící skript opravit.

4.4.2 Zjištění požadavků

Hodnocení bude prováděno na náhodně vygenerovaných datech. Hodnocení bude probíhat po proběhnutí modulů a na závěr budou hodnocení porovnána.

4.4.3 Specifikace hodnocení

Pro hodnocení byly vybrány míry **přesnosti**, protože tyto míry odpovídají tomu, na co se čistící modul a deduplikační modul zaměřují. Ostatní míry by také mohly být měřeny, ale vzhledem k tomu, že se jedná o principiálně stejné změny, nebyly by hodnoty nikterak odlišné.

Konkrétně byly vybrány tyto **míry**, které byly převzaty z teoretické části:

Míra syntaktické přesnosti pro jméno/příjmení klientů	
Funkce měření	A/B
Měřené elementy kvality	A = počet klientů, kteří mají jméno/příjmení syntakticky přesné (jméno/příjmení je k nalezení v doméně – číselníku)
	B = počet všech klientů

Tabulka 45: Míra syntaktické přesnosti pro ohodnocení klientů. Zdroj: (Vaníček, 2010)

Míra sémantické přesnosti klientů	
Funkce měření	A/B
Měřené elementy kvality	A = počet klientů, kteří jsou syntakticky přesní – označují jednoho unikátního klienta
	B = počet všech klientů

Tabulka 46: Míra sémantické přesnosti pro ohodnocení klientů. Zdroj: (Vaníček, 2010)

Stupnice měření jsou pro tyto míry zřejmé, jedná se o absolutní typ stupnice.

4.4.3.1 Kritéria pro vyhodnocení

Čištění a deduplikace bude považována za úspěšnou v případě, že oba typy vyhodnocených měr budou mít hodnotu přesnosti vyšší než 0,9.

4.4.4 Návrh hodnocení

Scénář postupu měření:

1. Vygenerování klientských dat
2. Nastavení koeficientu úprav v klientských datech
3. Nastavení koeficientu duplikace klientských dat
4. Měření přesnosti klientských dat
5. Spuštění dirty a duplikačního modulu
6. Měření přesnosti klientských dat
7. Spuštění clean a deduplikačního modulu
8. Měření přesnosti klientských dat

Jednotlivé kroky měření se opakují pro každé nastavení koeficientů 100 krát, aby výsledek měl větší vypovídající hodnotu. V tabulce níže jsou pak zaznamenány průměry ze sto naměřených hodnot. Kompletní tabulka naměřených hodnot je přiložena na CD.

Koeficienty úprav a duplikace postupně nabývají těchto hodnot: 0,1, 0,5 a 1.

5 Zhodnocení výsledků a doporučení

Bylo vygenerováno 1000 náhodných klientských záznamů a na těchto záznamech byly prováděny veškeré úpravy a měření. V příloze práce jsou zaznamenány průměry ze 100 měření pro každou kombinaci koeficientu úpravy a koeficientu duplikace.

Měřena byla syntaktická přesnost jména a příjmení a sémantická přesnost klientského záznamu.

5.1 Vyhodnocení výsledků syntaktických přesností

Z výsledků měření je patrné, že skript opraví v průměru 95 % poškozených příjmení a 94 % poškozených křestních jmen. Dále je vidět, že nezáleží na počtu úprav, skript opraví v průměru cca stejný poměr dat jak u křestních jmen, tak u příjmení.

Koeficient duplikace má na syntaktickou přesnost zanedbatelný vliv vzhledem k vlivu koeficientu úprav.

Čištění v případě syntaktických přesností můžeme považovat za úspěšné, protože všechny míry syntaktických přesností se zdaleka nepřiblížily k požadované hranici 0,9. Čištění podle zadaných kritérií splnilo očekávání.

Výsledky syntaktických přesností viz příloha práce.

5.2 Vyhodnocení výsledků sémantické přesnosti

U sémantické přesnosti je zřejmé, že na ni má vliv pouze koeficient duplikace a ne koeficient úprav. To je opačně než u syntaktických přesností, kde měl zase majoritní vliv koeficient úprav. V tabulce výsledků sémantické přesnosti přibyl ještě jeden typ hodnot, nerozpoznané duplikáty. Toto je průměrný počet duplikovaných klientů, kteří v rámci měření nebyly rozpoznány.

Výsledky sémantických přesností viz příloha práce.

Pro snadnější pochopení jsou v následující tabulce rozepsány jednotlivé rozpoznané/ nerozpoznané počty duplikátů. Jedná se o průměrné hodnoty:

	Koeficient duplikace	Počet klientů	Duplikovaný počet klientů	Rozpoznané duplikáty	Nerozpoznané duplikáty
1.	0.1	1000	100	99	1
2.	0.5	1000	500	482	18
3.	1	1000	1000	944	56

Tabulka 47: Počty duplikovaných klientů (ne)rozpoznaných. Zdroj: autor.

Z tabulky je vidět, že nárůst počtu nerozpoznaných duplikátů není konstantní, ale mocninný, což přisuzuji především tomu, že při velkém množství duplikací už nevznikají pouze duplikáty původních klientů, ale i duplikáty duplikátů, čímž se náročnost rozpoznání duplikátu zvyšuje.

Při malém množství úprav a malém množství duplikací je deduplikační skript schopen rozpoznat v průměru 99 % duplikovaných záznamů, což je velmi dobrý výsledek.

Pod pojmem malé množství úprav je myšleno 100 změn na 1000 klientů a pod pojmem malé množství duplikací je myšleno 100 duplikací na 1000 klientů.

Deduplikaci posuzovanou přes míru sémantické přesnosti můžeme také považovat za úspěšnou, protože ani zde se míra sémantické přesnosti zdaleka nepřiblížila k požadované hranici 0,9. Deduplikační skript tak splnil požadavky zadané před měřením.

V čistícím a deduplikačním skriptu jsou navíc vytvářeny navíc návrhy pro uživatele systému, které nebyly zohledněny v měření. Pokud by tyto návrhy byly zohledněny, klientská data tím by byla ještě zkvalitněna a výsledné přesnosti by byly ještě lepší.

Největším překvapením při implementaci bylo, že oproti deduplikačnímu skriptu je čistící skript náročnějším algoritmem na implementaci. Důvodem bylo to, že u deduplikace se dohledávaly „pouze" stejné záznamy a velmi zjednodušeně byl porovnáván počet shodných atributů, zatím co u čistícího skriptu byla složitější logika, kdy se program snažil dohledávat podobné záznamy a rozeznat původní správnou hodnotu.

Bylo velkou výhodou, že skript neopravoval data reálných uživatelů, ale data generovaná a upravená. Díky tomu bylo možno přesně změřit, kolik dat zůstalo po čištění neopraveno a jakým způsobem byla data nejprve poškozena a následně i opravena. Na základě této zpětné vazby bylo možno čistící skript jednoduše vyladit.

Dalším velkým benefitem při simulaci úprav/špinění dat byla možnost volitelně parametrem nastavit, do jaké míry mají být data poškozena. Tímto způsobem pak bylo možno simulovat různé úrovně kvality dat, které byly čištěny. Získat stejné množství různě kvalitních reálných a zároveň totožných dat, by nebylo jednoduše možné.

Jak jsem již zmínil, všechny skripty jsou parametrizovatelné a tím rozšiřitelné. Proto lze skripty rozšiřovat jak o další typy úprav, tak o další způsoby čištění. Co by ale mělo být parametrizováno pro každou typově rozdílnou sadu dat je parametrizace rozpoznání duplikací. Pro moje generovaná data jsem tuto parametrizaci postupně ladil, aby bylo rozpoznáno co nejvíce duplikátů. Povaha dat se ale může v různých systémech lišit, proto pro získání nejlepších možných výsledků je potřeba odladit parametrizaci pro každý systém zvlášť.

5.3 Shrnutí výsledků

Po odladění běhu skriptů nad malým počtem úprav (koeficient úprav 0,1) a duplikací (koeficient úprav 0,1) byly naměřeny syntaktické přesnosti 0,9939 pro křestní jména a 0,9949 pro příjmení. Tyto syntaktické přesnosti odpovídají 6 neopraveným křestním jménům a 5 neopraveným příjmením, kde do 1000 záznamů klientů bylo úpravami zaneseno 100 chyb. Pro sémantickou přesnost byla naměřena hodnota 0,9991, což odpovídá 1 neopravené duplikaci ze 100 vnesených duplikací pro 1000 záznamů klientů.

6 Závěr

Z teorie vyplynulo, že data je potřeba stále kontrolovat a čistit. Na tuto oblast je možné nahlížet z různých úhlů pohledů a dávat jednotlivým kritériím různou vážnost. Na základě mé zkušenosti je ale potřeba tyto procesy zakotvit již ve vizích společnosti, protože bez kvalitních požadavků manažerů není možné ukládat kvalitní data.

Ukládání dat v systémech je neustálý proces, kdy je snaha zaznamenat co nejlepším a nejvýstižnějším způsobem měnící se výsek reálného světa. V rámci kvality dat je následně potřeba stanovit, jaká data z reálného světa budou ukládána s jakou přesností, jak často atd. Z těchto požadavků se následně tvoří kritéria, která stanoví, co data musí splňovat, abychom je pro naše účely považovali za kvalitní.

Při samotné realizaci praktické části jsem si uvědomil, o jak komplexní oblast v informačních systémech se jedná a jak logika pro rozpoznání chybných dat je složitá. Uvědomuji si, že čistící skripty by bylo možné dále ladit a vylepšovat, aby počet opravených záznamů byl ještě větší.

Práce na analýze dat a jejich čištění byla pro mě velmi zajímavá a díky ní jsem si uvědomil přínos automatických skriptů pro zpracování dat. Bez těchto procesů by nebylo možné data zpracovávat a spravovat. Vzhledem k tomu, jaké objemy dat se v databázích uchovávají, je již nereálné si myslet, že by člověk byl schopen tato data zpracovávat ručně. O to více vidím velký přínos v hromadné opravě záznamů, která vede ke zvyšování kvality dat.

7 Seznam použitých zdrojů

Adelman, Sid, Moss, Larissa, Abai, Majid. 2005. *Data Strategy.* Crawfordsville : Addison-Wesley Professional, 2005. ISBN 0-321-24099-5.

Bancilhon, Francois, Delobel, Claude, Kanellakis, Paris. 1992. *Building an Object-Oriented Database System.* San Francisco : Morgan Kaufmann, 1992. ISBN 1-55860-169-4.

Boehm, Barry, Basili, Vic. 2001. Software defect reduction. *Software manangement.* Janury, 2001, Sv. I, 12.

Buxton, Stephen, Melton, Jim. 2006. *Querying XML.* San Francisco : Morgan Kaufmann, 2006. ISBN 1-55860-711-0.

Dalcin, Eduardo. 2005. *Data Quality Concepts and Techniques Applied to Taxonomic Databases.* Southampton : University of Southampton, 2005.

Elliott, Michael. 2015. *Oracle 12c Database Quickstart.* New York : Kobo, 2015. ISBN 978-1-310-89654-5.

English, Larry. 1999. *Improving Data Warehouse and Business Information Quality: Methods for Reducing Costs and Increasing Profits.* New York : John Wiley & Sons, Inc., 1999. str. 518. ISBN 978-0-471-25383-9.

Harrington, Jan. 2000. *Object-oriented Database Design Clearly Explained.* San Francisco : Morgan Kaufmann, 2000. ISBN 0-12-326428-6.

—. 2010. *SQL Clearly explained.* San Francisco : Morgan Kaufmann, 2010. ISBN 978-0-12-375697-8.

Hernandez, Stolfo. 1998. Real-world Data is Dirty: Data Cleansing and The Merge/Purge Problem. *Data Mining and Knowledge Discovery*. 2, 1998, Sv. II, 9.

Herout, Pavel. 2007. *Java a XML.* České Budějovice : KOPP, 2007. ISBN 80-7232-307-5.

Chapman, Arthur. 2005. *Principles and methods of data cleaning.* [Dokument] Copenhagen : Global Biodiversity Information Facility, 2005. ISBN 87-092020-04-6.

—. 2005. *Principles of Data Quality.* [Dokument] Copenhagen : Global Biodiversity Information Facility, 2005. ISBN 87-92020-49-6.

—. 2004. *Environmental Data Quality - b. Data Cleaning Tools.* [dokument] Campinas, Brazil : CRIA 57, CRIA 57, 2004.

Chodorow, Kristina. 2013. *MongoDB: The Definitive Guide.* Sebastopol : O'Reilly Media, 2013. ISBN 978-1-4493-4467-2.

Chrisman, Nicholas. 1991. The Error Component in Spatial Data. *Geographical information systems.* Summer, 1991, Sv. I, 1.

IAIDQ. 2015. IQ/DQ glossary. *International Association for Information and Data Quality.* [Online] IQ International, 19. July 2015. [Citace: 15. September 2016.] http://iaidq.org/main/glossary.shtml#R).

IT, solid. 2016. RANKING. *DB-ENGINES.* [Online] solid IT, 2016. [Citace: 20. September 2016.] http://db-engines.com/en/ranking.

Kostiha, František. 2012. *Měření a hodnocení kvality informačních systémů.* Praha : FF UK, 2012.

Laberge, Robert. 2012. *Datové sklady: agilní metody a business inteligence.* Brno : Computer Press, 2012. ISBN 978-80-251-3729-1.

Lacko, Luboslav. 2013. *Mistrovství v SQL Server 2012.* Brno : Computer Press, 2013. ISBN 978-80-2513-773-4.

Maletic, Jonathan, Marcus, Andrian. 2000. *Data Cleansing: Beyond Integrity Analysis.* Memphis : autor neznámý, 2000.

McGilvray, Danette. 2008. *Executing Data Quality Projects - Ten steps to Quality Data and Trusted Information.* Burlington : Morgan Kaufmann, 2008. ISBN 978-0-12-374369-5.

mongoDB. 2016. mongoDB documentation. *mongoDB.* [Online] MongoDB Inc., 2016. [Citace: 20. September 2016.] https://docs.mongodb.com/manual/.

Olson, Jack. 2003. *Data Quality the accuracy dimension.* San Francisco : Morgan Kaufmann, 2003. ISBN 1-55860-891-5.

O'Neil, Patrick. 1994. *Database - principles, programming, performance.* San Francisco : Morgan Kaufmann, 1994. ISBN 1-55860-219-4.

Oracle. 2016. Oracle Warehouse Builder Data Modeling, ETL, and Data Quality Guide. *Dokumentace oracle.* [Online] 2016. [Citace: 10. 03 2016.] http://docs.oracle.com/cd/E11882_01/owb.112/e10935/toc.htm.

Powell, Gavin. 2007. *Beginning XML Databases.* Washington : John Wiley & Sons, 2007. ISBN 978-0-471-79120-1.

Přibrský, Michal. 2012. Kvantifikovaný přístup k jakosti informačního zabezpečení pro podporu evaluace informačních technologií. *Systémová integrace.* 3, 2012.

Rahm, Erhard, Hong, Hai Do. 2000. Data Cleaning: Problems and Current Approaches. *Bulletin of the Technical Committee on Data Engineering.* 23, 2000, Sv. I, 4.

Redman, Thomas. 1996. *Data quality for the information age.* San Francisco : Artech House, 1996. ISBN 08-900-6883-6.

—. 2001. *Data Quality: The Field Guide.* Boston : Digital Press, 2001. ISBN 15-555-8251-6.

Roebuck, Kevin. 2011. *Data Quality: High-impact Strategies.* Brisbane : Emereo Pty Ltd, 2011. ISBN 17-430-4631-6.

Soley, Richard, Nour Ali, John Grundy, Bedir Tekinerdogan. 2016. *Software quality assurance.* Waltham : Morgan Kaufmann, 2016. ISBN 978-0-12-802301-3.

Vaníček, Jiří. 2010. *Information systems quality rating.* Praha : ČZU, 2010. ISBN 978-80-213-2062-8.

—. 2006. *Jak postupovat při hodnocení jakosti softwarových produktů.* [Dokument] Praha : ČZU, 2006.

—. 2004. *Měření a hodnocení jakosti informačních systémů.* Praha : ČZU, 2004. ISBN 80-213-1206-8.

Vaníček, Jiří, a kol. 2007. *Teoretické základy informatiky.* Praha : Kernberg Publishing, 2007. ISBN 978-80-903962-4-1.

Wang, Richard, Ziad, Mostapha, Lee, Yang. 2002. *Data Quality.* New York : Berlin, 2002. ISBN: 0-792-37215-8.

8 Příloha, výsledky běhů čistících skriptů

8.1 Získané výsledky pro syntaktickou přesnost

Měření	Popis	Koeficient úprav	Koeficient duplikace	Syntaktická přesnost jména	Diference
==	Po vygenerování	-	-	1	-
1.	Po úpravách	0,1	0,1	0,9814	-0,0186
	Po čištění			**0,9939**	0,0125
2.	Po úpravách	0,1	0,5	0,9812	-0,0188
	Po čištění			**0,9939**	0,0127
3.	Po úpravách	0,1	1	0,9816	-0,0184
	Po čištění			**0,9935**	0,0119
4.	Po úpravách	0,5	0,1	0,9136	-0,0864
	Po čištění			**0,9694**	0,0558
5.	Po úpravách	0,5	0,5	0,9113	-0,0887
	Po čištění			**0,9688**	0,0575
6.	Po úpravách	0,5	1	0,914	-0,0860
	Po čištění			**0,9715**	0,0575
7.	Po úpravách	1	0,1	0,8323	-0,1677
	Po čištění			**0,9414**	0,1091
8.	Po úpravách	1	0,5	0,8325	-0,1675
	Po čištění			**0,9425**	0,1100
9.	Po úpravách	1	1	0,8357	-0,1643
	Po čištění			**0,9424**	0,1067

Tabulka 48: Výsledky měření syntaktická přesnost jména. Zdroj: autor.

Měření	Popis	Koeficient úprav	Koeficient duplikace	Syntaktická přesnost příjmení	Diference
==	Po vygenerování	-	-	1	-
1.	Po úpravách	0,1	0,1	0,9827	-0,0173
	Po čištění			**0,9949**	0,0122
2.	Po úpravách	0,1	0,5	0,984	-0,0160
	Po čištění			**0,9952**	0,0112
3.	Po úpravách	0,1	1	0,9843	-0,0157
	Po čištění			**0,9958**	0,0115
4.	Po úpravách	0,5	0,1	0,9175	-0,0825
	Po čištění			**0,9745**	0,0570
5.	Po úpravách	0,5	0,5	0,9221	-0,0779
	Po čištění			**0,9764**	0,0543
6.	Po úpravách	0,5	1	0,9261	-0,0739
	Po čištění			**0,9768**	0,0507
7.	Po úpravách	1	0,1	0,8443	-0,1557
	Po čištění			**0,9485**	0,1042
8.	Po úpravách	1	0,5	0,8543	-0,1457
	Po čištění			**0,9523**	0,0980
9.	Po úpravách	1	1	0,8596	-0,1404
	Po čištění			**0,9535**	0,0939

Tabulka 49: Výsledky měření syntaktická přesnost příjmení. Zdroj: autor.

8.2 Získané výsledky pro sémantickou přesnost

Měření	Popis	Koeficient úprav	Koeficient duplikace	Sémantická přesnost klientů	Diference (Nerozpoznané duplikáty)
==	Po vygenerování	-	-	1	-
1.	Po úpravách	0,1	0,1	0,9091	-0,0909
	Po čištění			**0,9991**	0,0900 (1)
2.	Po úpravách	0,1	0,5	0,6667	-0,3333
	Po čištění			**0,9836**	0,3169 (17)
3.	Po úpravách	0,1	1	0,5	-0,5000
	Po čištění			**0,9491**	0,4491 (54)
4.	Po úpravách	0,5	0,1	0,9091	-0,0909
	Po čištění			**0,999**	0,0899 (1)
5.	Po úpravách	0,5	0,5	0,6667	-0,3333
	Po čištění			**0,9826**	0,3159 (18)
6.	Po úpravách	0,5	1	0,5	-0,5000
	Po čištění			**0,9464**	0,4464 (57)
7.	Po úpravách	1	0,1	0,9091	-0,0909
	Po čištění			**0,9987**	0,0896 (1)
8.	Po úpravách	1	0,5	0,6667	-0,3333
	Po čištění			**0,9817**	0,3150 (19)
9.	Po úpravách	1	1	0,5	-0,5000
	Po čištění			**0,9459**	0,4484 (57)

Tabulka 50: Výsledky měření sémantická přesnost klientů. Zdroj: autor.

9 Seznamy

9.1 Obrázky

9.2 Tabulky

9.3 Zdrojové kódy