VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA PODNIKATELSKÁ
FACULTY OF BUSINESS AND MANAGEMENT

ÚSTAV INFORMATIKY
INSTITUTE OF INFORMATICS

NÁVRH DATOVÉHO SKLADU
DESIGN OF DATA WAREHOUSE

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. David Szkuta**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Petr Dydowicz, Ph.D.**
SUPERVISOR

BRNO 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA PODNIKATELSKÁ

Zadání diplomové práce

Ústav:	Ústav informatiky
Student:	**Bc. David Szkuta**
Studijní program:	Systémové inženýrství a informatika
Studijní obor:	Informační management
Vedoucí práce:	**Ing. Petr Dydowicz, Ph.D.**
Akademický rok:	2017/18

Ředitel ústavu Vám v souladu se zákonem č. 111/1998 Sb., o vysokých školách ve znění pozdějších předpisů a se Studijním a zkušebním řádem VUT v Brně zadává diplomovou práci s názvem:

Návrh datového skladu

Charakteristika problematiky úkolu:

Úvod
Vymezení problému a cíle práce
Teoretická východiska práce
Analýza problému a současné situace
Vlastní návrh řešení, přínos práce
Závěr
Seznam použité literatury

Cíle, kterých má být dosaženo:

Cílem práce je navržení datového skladu podle potřeb společnosti. Před samotným návrhem však bude provedena analýza současného stavu. Součástí analýzy bude také průzkum dostupných technologií.

Základní literární prameny:

BASL, J. a R. BLAŽÍČEK. Podnikové informační systémy. Podnik v informační společnosti. Praha: Grada, 2008. 283 s. ISBN 978-80-247-2279-5.

MOLNÁR, Z. Automatizované informační systémy. Praha: Strojní fakulta ČVUT, 2000. 126 s. ISBN 80-01-02269-2.

MOLNÁR, Z. Efektivnost informačních systémů. Praha: Grada Publishing, 2000. 142 s. ISBN 80-716--410-X.

ŘEPA, V. Analýza a návrh informačních systémů. Praha: Ekopress, 1999. 403 s. ISBN 80-86119--3-0.

SODOMKA, P. a H. KLČOVÁ. Informační systémy v podnikové praxi. Brno: Computer Press, 2010. 501 s. ISBN 978-80-251-2878-7.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2017/18

V Brně dne 28.2.2018

L. S.

doc. RNDr. Bedřich Půža, CSc. ředitel	doc. Ing. et Ing. Stanislav Škapa, Ph.D. děkan

Abstrakt

Tato diplomová práce se zabývá návrhem datového skladu pro ukládání událostí z mobilní aplikace. Cílem bylo navržení datového skaldu, jenž poslouží jako alternativa k současně používanému systému. V práci jsou vysvětleny pojmy, jež se v rámci celé práce vyskytují, ty jsou zaměřeny především na problematiku datových skladů. Dále je provedena analýza pro posouzení současně používaného systému a také dostupných řešení v oblasti datových skladů a ETL služeb. V závislosti na výsledcích analýzy je vybráno vhodné řešení, které je naimplementováno a odzkoušeno.

Abstract

This diploma thesis deals with the design of a data warehouse that stores events created in the mobile app. The goal was to design an alternative to the current solution. The thesis explains concepts, mainly data warehouse terminology, which are used in subsequent chapters. An analysis of the current solution is conducted, and as well as research of available data warehouse and ETL services. Based on the results of the analysis, a suitable new solution is chosen, implemented, and tested.

Klíčová slova

Datový sklad, ETL proces, mobilní aplikace, SQL, Python

Keywords

Data Warehouse, ETL process, mobile application, SQL, Python

Bibliografická citace

SZKUTA, D. Návrh datového skladu. Brno: Vysoké učení technické v Brně, Fakulta podnikatelská, 2018. 86 s. Vedoucí diplomové práce Ing. Petr Dydowicz, Ph.D..

Čestné prohlášení

Prohlašují, že předložená diplomová práce je původní a zpracoval jsem ji samostatně. Prohlašují, že citace použitých zdrojů je úplná, že jsem ve své práci neporušil autorská práva (ve smyslu Zákona č. 121/2000 Sb., o právu autorském a o právech souvisejících s právem autorským).

V Brně dne 20. 5. 2018 ……

Poděkování

Tímto bych rád poděkoval svému vedoucímu práce Ing. Petru Dydowiczovi Ph.D., za jeho odbornou pomoc při psaní této diplomové práce.

Zároveň bych také chtěl poděkovat spolupracovníkům a rodině, rovněž za jejich odborné připomínky a rady.

Obsah

Úvod

Data jsou považována za jedno z největších bohatství společnosti, jelikož se na jejich podkladech rozhoduje v rámci všech úrovní společnosti. Proto je v zájmu každé společnosti zajistit pro rozhodovací potřeby data s nejlepší možnou dostupností a kvalitou.

Současné databázové systémy, zejména pak datové sklady, umožňují pracovat s velkými objemy dat, čímž společnostem pomáhají k pochopení chování jejich zákazníků, optimalizaci produktu a nabídky, ale také identifikaci možných problémů v závislosti na nastavených metrikách.

Nynější situace, kdy moderní technologie zapříčinily rozmach prodeje zboží a služeb skrz webové stránky nebo mobilní aplikace je prakticky nutností tyto kanály sledovat a zajistit si tak přehled o svých uživatelích. Technologické společnosti tento trend a požadavky zaregistrovaly, a poskytují tyto produkty, ať už jako jednu z mnoha služeb ve svém portfoliu, což je například Google Analytics, nebo se soustředí čistě na něj. Zmíněné produkty jsou jednoduché na implementaci a zajišťují uspokojení základních požadavků. V případě potřeby komplexnějšího systému je jedním z řešení kombinace datového skladu, který je provozován v cloudu. Zákazník tedy využívá jen datový sklad, ale o její údržbu a dostupnost se stará poskytovatel služby. Ideální je pak kombinace s některou z ETL služeb, jež data pomocí svého SDK sbírá a následně upravuje a nahrává do datového skladu.

Cíle práce, metody a postupy zpracování

Cílem této práce je navržení datového skladu pro společnost XYZ, jenž by v první fázi měl pokrýt potřeby na zaznamenávání událostí v mobilní aplikaci, která slouží jako jeden z kanálů, kde společnost nabízí své produkty.

V současné době firma používá řešení, které by se dalo považovat jako alternativa datového skladu, avšak v analýze současného stavu pomocí SWOT analýzy bude určeno, zda současně používané řešení vyhovuje. Stav současného řešení bude zohledněn zejména v kontextu současně dostupných technologií a produktů, jejichž srovnání bude také součástí stejné kapitoly.

Kapitola návrhu řešení bude následně obsahovat návrh samotného datového skladu. Současně se v této kapitole objeví také finanční informace o nákladnosti celého projektu a také přibližné náklady, které budou potřeba na samotný provoz nového nebo upraveného řešení.

Jako podklad pro čtenáře, kteří nemají zkušenosti s touto problematikou, bude před kapitolami o analýze a návrhu zpracována teoretická část, která by měla pomoci s orientací také těmto čtenářům.

1 Teoretická východiska práce

V této kapitole budou obsaženy teoretická východiska a pojmy, které se v práci následně budou vyskytovat.

1.1 Datový sklad

Datový sklad slouží k shromažďování a uchovávání dat po co možná nejdelší časové období. Uložená data by měla poskytovat aktuální a přesné odpovědi v co nejkratším časovém úseku. Dle Billa Inmona byl datový sklad definován následovně: "Datový sklad je podnikově strukturovaný depozitář objektově orientovaných, integrovaných, časově proměnných, historických dat použitých pro získávání informací a podporu rozhodování. V datovém skladu jsou uložena atomická a sumární data [1, s. 22]"

V závislosti na definici musí datový sklad splňovat následující požadavky:

- Subjektovou orientaci – Data se zapisují do datového skladu v závislosti na předmětu zájmu. Data jsou kategorizována podle subjektu (například zákazník, dodavatel, výrobek apod.). Není vhodné subjekty nahrazovat aplikacemi, ve kterých byly data vytvořeny (software pro prodej, fakturaci, personalistiku atd.)
- Integrovanost – Podmínkou pro správnou funkci datového skladu je jednotnost dat a integrovanost. Pointou této podmínky je ukládání dat týkající se konkrétního předmětu pouze jednou. Proto musí být vedena jednotná terminologie a jednotky veličin. Riziko je vyšší v případě, kdy je zdrojem několik, vzájemně nezávislých, systémů. Tento požadavek by měl zajistit proces ETL.
- Časovou variabilitu – Data se ukládají do datového skladu jako série snímků, přičemž každý z nich reprezentuje určitý časový úsek. Z toho vyplývá, že data jsou platná pouze pro určitý časový moment (časový snímek). V operačních databázích jsou data uložena po výrazně kratší časové období (dny, týdny až měsíce) než je tomu u datových skladů, kde jsou dostupná data za několik let. Jakmile je v datovém skladu zaznamenán snímek dat z operační databáze, pak nemohou být tato data dále modifikována.

- Neměnnost – V operačně transakčních databázích jsou s daty prováděny všechny možné operace (vytváření/vkládání, modifikace nebo mazání). Datový sklad však jen data vkládá a následně k nim přistupuje, neprobíhají tedy operace pro odstraňování záznamů nebo jejich modifikace. Také proto je v datových skladech nepotřebná normalizace dat [1].

1.1.1 Rozdíly mezi produkční databází a datovým skaldem

Tabulka č. 1: Rozdíly mezi produkční databází a datovým skaldem.
(Zdroj: 1)

Metrika	Produkční databáze	Datový sklad
Čas odezvy	V sekundách	Sekundy až hodiny
Operace	DML (Data manipulation lan)	Primárně pro čtení
Původ dat	30–60 dní	Série snímků za časový úsek
Organizace dat	Podle aplikace	Podle předmětu, času apod.
Velikost	Malá až velká	Velká až velmi velká
Zdroj dat	Operační a interní	Operační, interní a externí
Činnost	Procesy	Analýza

Datový sklad jako celek může být tvořen datovými trhy, jež jsou podmnožinou datového skladu a jsou vytvářeny například pro divize v rámci firmy v rámci organizace, geografickou lokalitu nebo určité oblasti podnikání. Často jsou vytvářeny z důvodu velké náročnosti projektu tvorby datového skladu. Proto se v některých případech tvorby datového skladu postupuje právě touto cestou, kdy je společnost rozdělena na podmnožiny, které se následně integrují do datového skladu. Výhodou tohoto řešení je menší ekonomické zatížení [1].

Avšak je možné postupovat i opačně, kdy se vytvoří centrální datový sklad, který se následně rozdělí do několika datových trhů. Výhodou jsou pak menší náklady na provoz a údržbu [1].

1.1.2 Metody budování datového sklady

Existují dvě nejznámější a nejpoužívanější metody budování datového skladu:

- Metoda velkého třesku – Jde o realizaci datového skladu pomocí jediného projektu. Největší riziko spojené s touto metodou je časová náročnost realizace, kdy se během procesu realizace mohou dokonce zastarat technologie, které bylo původně zamýšleno použít nebo požadavky zákazníka. Dále je pak delší časový interval návratnosti než v druhé metodě. V této metodě se používá následující skladba etap:
  1) analýza podniku,
  2) vytvoření datového skladu,
  3) vytvoření přístupu přímo nebo pomocí datových trhů.
- Přírůstková metoda – Na rozdíl od předchozí metody se datový sklad buduje postupně (po jednotlivých etapách). Jako první se vytvoří pár datových tržišť, přičemž se pomalu škálují dle potřeb podniku. Již první podmnožiny datového skladu se spustí a zpřístupní koncovým uživatelům, čímž se zároveň testuje funkcionalita. Následně se začínají nabalovat další tržiště až postupně vznikne konečný datový sklad. Výhodami je budování skladu v souladu s potřebami uživatelů, je snadno rozšiřitelný a vykazuje rychlejší návratnost vložených prostředků [1].

Budování datového skladu pomocí přírůstkové metody má dvě varianty:

1) Směrem “shora dolů” - v této variantě je vytvořen konceptuální model datového skladu, přičemž se stanoví také hierarchie předmětných oblastí. Následně se sestaví modely jednotlivých předmětných oblastí. Varianta se vyznačuje poměrně rychlou implementací datových trhů a vysokými vstupními náklady.
2) Směrem “zdola nahoru” - v této variantě se nejdříve vybudují datové trhy předmětných oblastí v rámci struktury datového skladu. Jelikož se konceptuální model varianty odvíjí od zdrojových systémů, je případná rozšiřitelnost problematická [1].

1.1.3 Schéma datového skladu

Datové sklady jsou tvořeny tabulkami dvou typů. Prvním druhem je takzvaná tabulka faktů, jenž obsahují číselné hodnoty. Druhým je pak tabulka dimenzí, jenž poskytují kontext pro tabulky faktů.

V závislosti na přístupu při modelování schématu datového skladu se používá především schéma hvězdy nebo sněhové vločky [1], [2].

1.1.3.1 Tabulka faktů

Jak již bylo zmíněno obsahuje metrické údaje. Zpravidla se jedná o největší tabulky v databázi, jelikož každý záznam (řádek tabulky) odpovídá jedné události (například nákup, zapnutí aplikace apod.). Hlavním předpokladem optimální funkčnosti je mít záznamy v rámci jedné tabulky faktů na stejné úrovni. Je doporučeno navrhovat tabulky faktů na základě událostí na nejnižších úrovních, aby se předešlo případné následné duplikaci. V ideálním případě by tabulka faktů měla obsahovat sadu cizích klíčů odkazující na primární klíče v tabulkách dimenzí a sadu metrik. Primární klíč u tabulky faktů je obvykle v podobě složeného primárního klíče, jenž je dán složením přítomných cizích klíčů [2].

1.1.3.2 Tabulka dimenzí

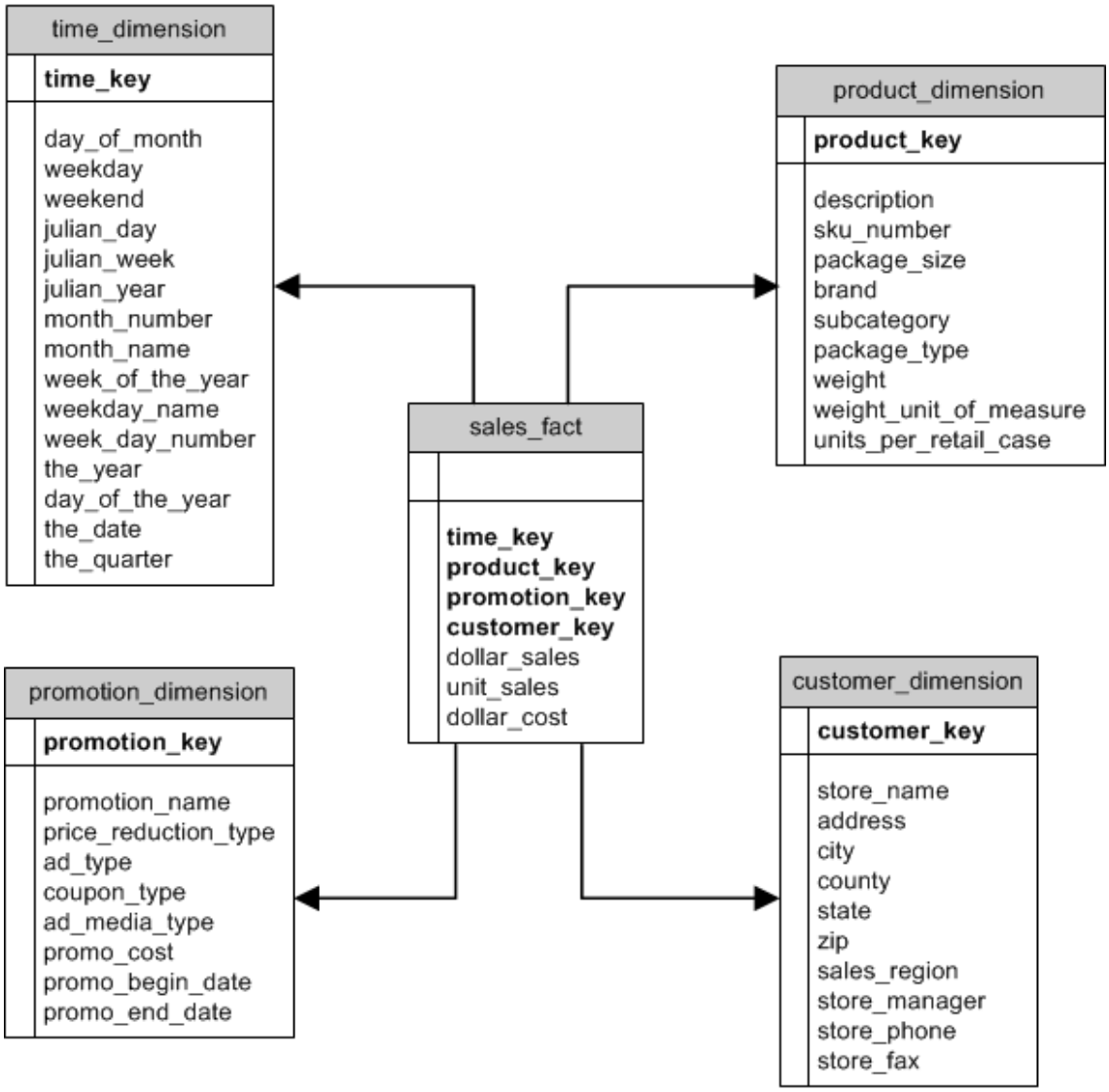
Tabulky dimenzí slouží jako kontext (popis) pro tabulky faktů. Poskytují informace jako – kdo, kdy, kde apod. Každý záznam v tabulce dimenzí má svůj jedinečný identifikátor, který je na rozdíl od tabulky faktů nejčastěji jednoduchý. Nejčastěji se vyskytují časové, geografické, produktové/kategorické a zákaznické dimenze [2].

1.1.3.3 Schéma hvězdy

Schéma hvězdy (Star Schema, Star Join) označuje případ, kdy je každý proces reprezentován dimenzionálním modelem obsahujícím tabulku faktů pro každou událost, která je obklopena tabulkami dimenzí, jež poskytují kontext. Schéma je symetrické a jednoduché, což zjednodušuje navigaci v datech a snižuje riziko chyb. Díky jednoduchosti jsou dotazy rychlejší, jelikož není nutné procházet spoje mezi tabulkami, což umožňuje databázi fungovat efektivněji, a celé schéma je snadno rozšiřitelné o další dimenze [2].

1.1.3.4 Souhvězdí

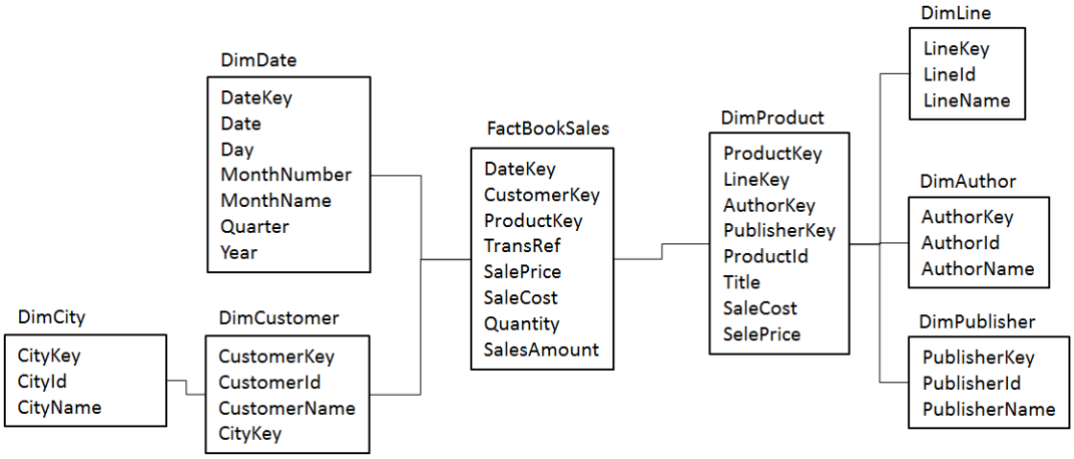
Schéma souhvězdí (Constellation Schema) vychází ze schématu hvězdy. Toto schéma v sobě zahrnuje několik schémat hvězd. Souhvězdí obsahuje minimálně dvě tabulky faktů, které spolu sdílejí dimenze [2].



Obrázek č. 1: Příklad schématu hvězdy.
(Zdroj: 4)

1.1.3.5 Schéma sněhové vločky

Schéma sněhové vločky (Snowflake Schema) vychází ze schématu hvězdy. Schéma funguje tak, že ve schématu hvězdy normalizuje (celé nebo jen část), čímž může mít jedna větev dimenzí více úrovní. Schéma dává smysl především v případech s vyšší pravděpodobností dodatečných změn a náročností údržby. Nevýhodou je však degradace dotazovacího výkonu z důvodu většího počtu (kaskádových) spojení. Z těchto důvodů se obecně doporučuje, pokud to není nezbytně nutné, nepoužívat schéma sněhové vločky, ale upřednostnit schéma hvězdy [3].



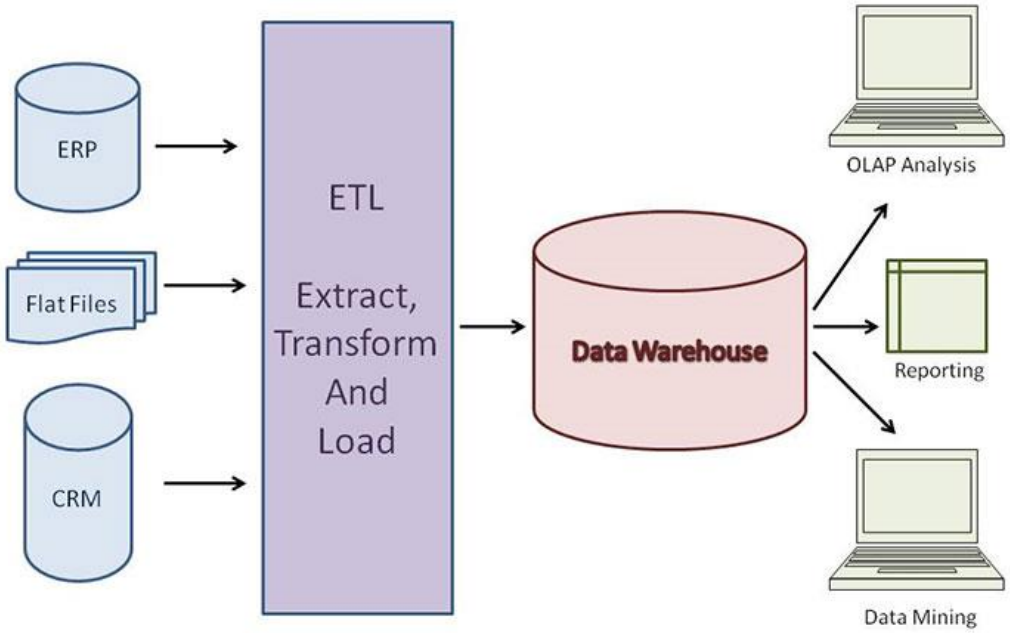
Obrázek č. 2: Příklad schématu sněhové vločky.
(Zdroj: 5)

1.2 ETL

Pod zkratkou ETL se skrývá třífázový systém. Konkrétně se jedná o extrakci (Extract), transformaci (Transformation) a načtení (Loading). Hlavním úkolem systému je do datového skladu doručit data v co možná nejlepší kvalitě [2].

Proces ETL je složen z následujících fází:

- Extrakce – Jedná se o první krok procesu. Cílem této fáze je shromáždit data z požadovaného zdroje a přesunout je do místa, kde se nad nimi bude dále manipulovat (dočasné úložiště). Již v tomto kroku jsou data již součástí datového skladu.
- Transformace – Po nahrání dat je potřeba data převést do podoby, která je relevantní pro cílový datový sklad. Nejčastěji se jedná o operace, při kterých se opravují chybějící hodnoty, standardizuje formát numerických nebo textových hodnot, provádí výpočet hodnot nebo agregace a vypořádává se s duplicitami.
- Načtení – Jedná se o poslední krok ETL systému. Cílem tohoto kroku je nahrání připravených dat do připraveného schématu datového skladu [2].



Obrázek č. 3: Grafické znázornění ETL procesu.
(Zdroj: 11)

1.3 SQL

SQL (zkrácením původního názvu SEQUEL) je jazyk, který se používá k manipulaci v relačních databázích. Ačkoli byl vytvořen standard, je syntaxe jazyka SQL odlišná v závislosti na použitém databázovém systému (například Microsoft – T-SQL, MySQL – SQL/PSM nebo PostgreSQL – PL/pgSQL) [6].

Jazyk SQL je rozdělen do tří částí, přičemž každá z nich se věnuje práci s odlišnými prvky v databázi [6].

- SQL schema statements – Používají se k definici datových struktur, ve kterých jsou data uložena.
- SQL data statements – Slouží k manipulaci s datovými strukturami vytvořenými pomocí předchozího kroku.
- SQL transaction statements – Jejich úkolem je provádění transakcí [6].

SQL je svázáno s relačními databázemi, jelikož jsou data uložená v tabulkách a rovněž výsledky dotazů jsou tabulky. Všechny prvky vytvoření pomocí SQL schema

statement jsou uloženy ve speciální relaci tabulek zvaných datový slovník, někdy jsou označovány také jako metadata [6].

1.4 Databáze

Databáze je definována pokaždé trochu jinak v závislosti na vývojáři, jenž databázový systém vyvíjí. Obecně lze však databázi definovat jako kolekci vzájemně souvisejících položek, jež jsou spravovány jako jedna položka [7].

Označení databáze je pouze zkráceným označením pro systém řízení báze dat (zkráceně SŘDB). Tento systém provádí správu, údržbu a uspořádání databáze samotné [7].

1.4.1 Tabulka

Pod označením tabulka se skrývá hlavní jednotka pro ukládání dat v rámci databáze. Tabulka, taktéž někdy označována jako relace, má zpravidla dva rozměry, což jsou řádky a sloupce. Tabulka jako taková představuje objekt, což může být například věc nebo člověk, a každý řádek v rámci této tabulky představuje právě jeden výskyt entity. V rámci celé databáze jsou pak tabulky spojeny pomocí relačních vazeb [7].

Každá tabulka obsahuje množinu atributů, které jsou reprezentovány sloupci tabulky. Zpravidla se jedná o atomický údaj, který má vlastní datový typ. Datové typy se mohou měnit v závislosti na možnostech databázového systému avšak obecně bývají podporovány datové typy jako například:

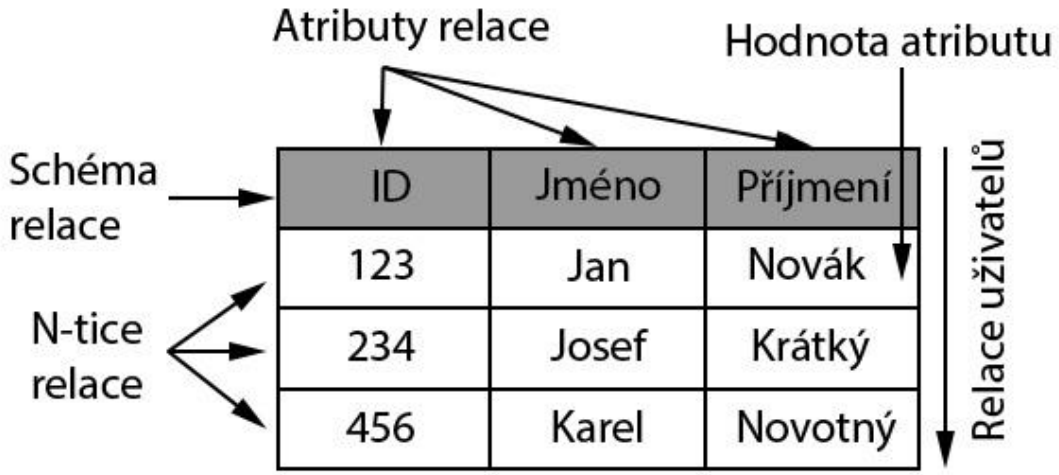
- Textové – char a varchar.
- Číselné – integer, double a float.
- Čas – datetime, date, timestamp (časová známka).
- Ostatní – boolean [7].

Relační vazby mají tzv. integritní omezení, která omezují kardinalitu vztahu pro vazby entit. Integritní omezení jsou následující:

- 1:1 - „Vztah 1:1 nám říká, že vždy jedné n-tici relace odpovídá jedna nebo žádná n-tice jiné relace.“ [7, s. 31]

- 1:N – „Vztah 1:N nám říká, že vždy jedné n-tici relace odpovídá jedna nebo více n-tic jiné relace.“ [7, s. 31]
- N:M – „Vztah N:M nám říká, že obecně několika n-ticím relace odpovídá více n-tic jiné relace.“ [7, s. 32]

Vztah N:M není možné v rámci návrhu databáze tolerovat a je nutné jej vyřešit za pomocí dekompozice vztahu [7].



Obrázek č. 4: Relace podle teorie relací.
(Zdroj: 7)

1.4.2 Typy databází

Vše, co bylo v rámci této podkapitoly doposud napsáno platí především pro tzv. relační databáze (někdy také označováno jako SQL databáze), které jsou více rozšířené a patří mezi ně databázové systémy jako PostgreSQL, MySQL, Oracle DB, Microsoft SQL Server a další. Ovšem mimo ně se lze v praxi setkat s tzv. ne-ralačními databázemi (Non-relation, NoSQL). NoSQL databáze nepoužívají objekty jsou jsou tabulky, namísto nich jsou dokumentově-orientované, což dovoluje snadno ukládat i nestrukturovaná data jako jsou obrázky nebo videa. Data jsou uložena v JSON souborech [8].

1.4.3 Index

Pokud je do tabulky vložen nový záznam, ve výchozím stavu nejsou tyto záznamy řazeny (například podle numerické hodnoty identifikátoru), nýbrž jsou nové záznamy

zapisovány do volného místa v souboru, kde má každá tabulka vyhrazené místo. Pokud je následně nutné vyhledat v neindexované tabulce nějaký záznam dle podmínky v níž se hodnota sloupce rovná jedné, musí databázový systém procházet všechny řádky dané tabulky. Ovšem pokud je vytvořen vzestupný index na daný sloupce, databázový systém odvodí na které množině řádků v tabulce hledat, což je mnohem efektivnější a rychlejší řešení, přičemž platí, že úspora času je přímo úměrná počtu řádků v tabulce [6].

Kromě "běžných" indexů jsou dostupné také unikátní indexy. Pokud je takový index aplikován na nějaký ze sloupců, pak je při následném vkládání hodnot zabráněno vložit v daném sloupci stejnou hodnotu, která již obsažena je. Jde o podobný mechanismus, který kontroluje duplicitu primárních klíčů do tabulky, při vkládání nových záznamů [6].

1.4.4 Metadata

Jednoduše jsou metadata označována jako "data o datech". Slouží pro ukládání záznamů o databázových objektech. Pro jednu vytvořenou tabulku obsahující indexy jsou poté v tabulce s meta daty obsaženy informace jako:

- název tabulky,
- názvy sloupců a jejich datové typy,
- sloupec nebo sloupce jež jsou primárním klíčem,
- sloupec nebo sloupce jež jsou cizím klíčem,
- název indexu, jejich typ, sloupce, jež se index týká a jejich způsob řazení [6].

Souhrnně jsou tyto informace označovány pojmy datový slovník nebo systémový katalog. Data ze slovníku se používají jako ověřovací mechanizmus při dotazech a můžou být modifikována pouze příslušnými nástroji jako je příkaz *ALTER TABLE* [6].

Přehled o všech objektech v rámci databáze jsou uchovávány v pohledech *information_schema* [6].

1.5 CSV

CSV (Comma-separated Values – Hodnoty oddělené čárkami) je soubor jenž je hojně využíván k přenosu dat mezi různými systémy nebo programy a definuje jej MIME

typ text/csv. Nejlépe pracuje s daty, jenž mají tabulkovou sktrukturu. Hlavní výhodou formátu je velká rozšířenost [9], [10].

1.6 JSON

JSON (JavaScript Object Notation) je rovněž jako CSV soubor, jenž slouží k přenosu dat mezi systémy a programy. A definuje jej MIME typ application/json. Data můžou být obsažena v polích nebo objektech. Každý objekt má svůj název a může obsahovat hodnoty typu objekt, pole, řetězec, číslo, boolean (True/False) a null [12].

1.7 Klastr

Klast (cluster) je skupina počítačů, jenž jsou nejčastěji propojeny pomocí lokální počítačové sítě, vystupující jako jeden stroj (systém). Důvodem vytváření klastrů je zvýšení výkonu a zamezení nedostupnosti z důvodu chyb. Kluster má vždy přiděleny jeden virtuální nebo fyzický stroj, který je tzv. Master (head node = hlavní uzel), který vykonává hlavní úkoly. Ostatní připojené uzly čekají až jím hlavní uzel přidělí úlohy na zpracování. Výsledky jsou v co nejkratším možném čase zpětně vráceny hlavnímu uzlu. Výhodou je, že hlavní uzel může přidělovat úlohy tak, aby se zpracovávaly paralelně [13]

1.8 Amazon S3

Amazon Simple Storage Service (S3) je cloudové úložiště společnosti Amazon Web Services (AWS). Data o architektuře úložiště nejsou bohužel veřejně dostupná, avšak obecně se jedná o úložiště založené na objektovém ukládaní dat (podobně jako pevný disk v počítači) [14].

Koncepce úložiště Amazon S3 je založena na následujících prvcích:

- Bucket (kyblík) - Bucket je kontejner pro objekty uložené v Amazon S3. Každý objekt musí být obsažen v některém z kontejnerů. Následná adresa objektu je vždy ve tvaru http://kontejner.s3.amazonaws.com/slozka/objekt.txt.
- Objects (objekty) - objekty jsou základní entitou uloženou v Amazon S3. Objekty jsou složeny ze samotných dat reprezentujících objekt a metadat. Metadata je množina párů název-hodnota popisující objekt s informacemi o poslední úpravě

nebo typu obsahu. Do metadat objektu je možné vytvořit také vlastní (uživatelská) metadata. Objekt je v rámci bucketu jednoznačně identifikován pomocí klíče (jména) a ID verze. Verzování umožňuje obnovit smazané nebo aktualizované (přepsané) objekty.

- Keys (klíče) - jak již bylo zmíněno, jedinečnou identifikaci objetu v rámci bucketu má na starosti klíč spolu s identifikátorem verze. Pokud se k těmto dvoum atributům dosadí také název bucketu, pak se jedná o jedinečný identifikátor v rámci celého úložiště Amazon S3. Informace o bucketu a klíči lze opět vyčíst z URL adresy, kdy v případě http://kontejner.s3.amazonaws.com/slozka/objekt.txt je kontejner název kontekneru a řetězec slozka/objekt.txt tvoří klíč.
- Regions (regiony) - jedná se o geografický region, kde bude vytvořen bucket. Objekty uložené v určitém regionu jej nikdy neopustí, přičemž může nastat výjimka, kdy je objekt explicitně přesunut do jiného regionu [14].

Amazon S3 Data Consistency Model (Model konzistence dat) - S3 poskytuje read-after-write konzistenci pro PUT metody nových objektů v bucketu pro všechny regiony s jedním varováním. Varování je při HEAD/GET HTTP metodách žádájících klíč před vytvořením samotného objektu. V takovém případě S3 poskytne zmiňovanou konzistenci read-after-write. Taktéž aktualizace klíče jsou atomické, takže pokud je použita metoda PUT pro tak následné čtení může vrátit stará nebo aktualizovaná data, ale nikdy nevrátí poškozená nebo částečná. Jelikož jsou data redundatně rozeseta po různých data centrech zabere nějakou chvíli, než se aktualizace dat zreplikují také na redundantní úložiště [14].

Úložiště Amazon S3 zákazníkům umožňuje vybrat typ úložiště (Storage Class) podle jeho potřeb, přičemž je možnost výběru z následujících variant:

- Standard – pro případy, kdy hraje roli výkon a je datům přistupováno často. Jedná se o výchozí možnost.
- Standard IA – zkratka IA označuje infrequent access (občasný přístup). Určeno pro případy, kdy není vyžadován častý přístup k datům, jsou data uložena na delší období, ale přesto stále potřebují vysoký výkon.
- Glacier – určená pro uchovávání dat s občasným přístupem k datům. Objekty uložené v rámci toho typu nejsou dostupné v reálném čase, jelikož objekt se musí

nejdříve obnovit a až poté je možno k němu přistupovat. Jedná se o finančně nenáročné řešení. Glacier se využívá jako doplněk k ostatním typům.

- Reduced Redundancy – má nižší úroveň redundance, než je tomu ve standardu. Určeno pro nekritická data [14].

Amazon S3 podporuje rozhraní REST API, jež umožňuje komunikaci mezi úložištěm a různými programovacími jazyky [14].

1.9 Massively Parallel Processing

Massively Parallel Processing (MPP – masivní paralelní zpracovávání) je technologie určená ke zpracovávání obrovských datových sad z datových skladů pomocí mnoha procesorů. Technologie vzešla z rostoucího trendu ohledně big data, kdy konvenční systémy měli se zpracováváním obrovského množství dat problém, jelikož k takovému účelu nebyly ani zamýšleny. V současné době je technologie součástí cloudových datových skladů jako je Amazon Redshift nebo Azure SQL Data Warehouse [15].

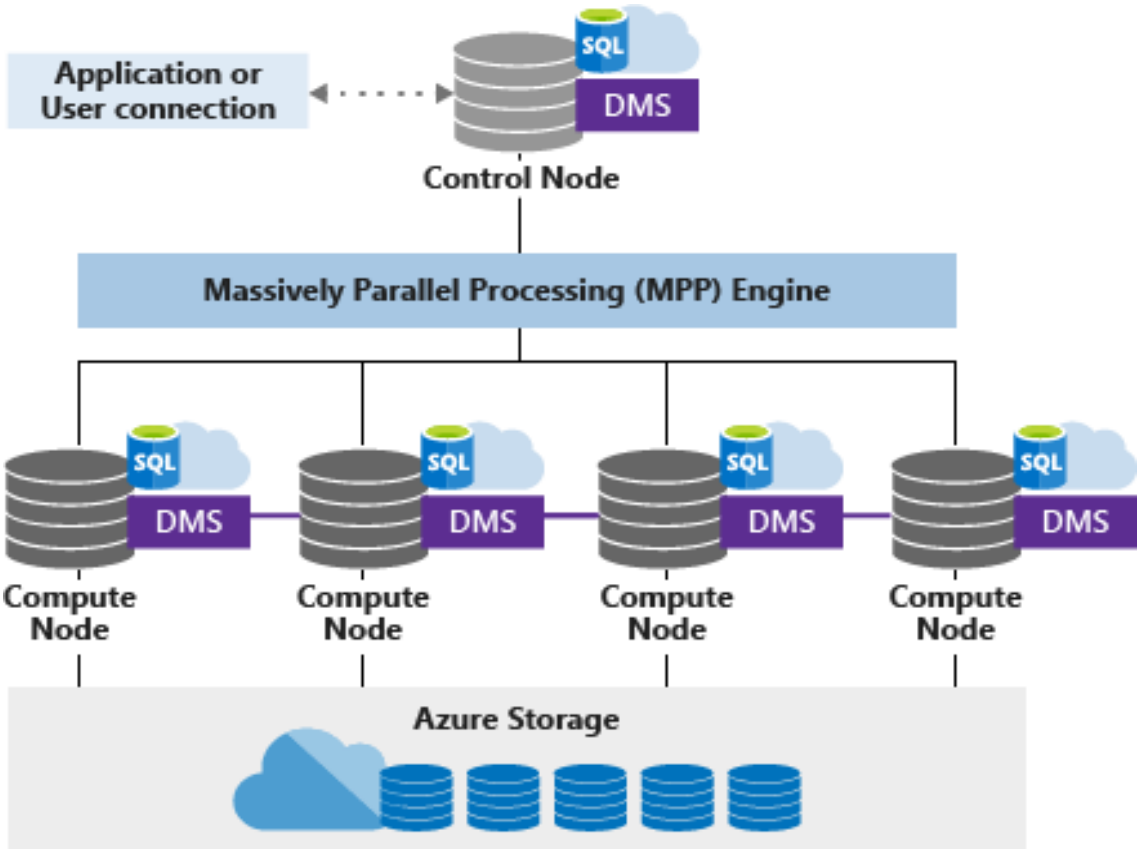
MPP funguje tak, že velký datový set je rozdělen na menší části a následně je každá část přiřazena jednomu procesoru, který má vyhrazen vlastní operační systém a paměť. To umožňuje, aby každý procesor pracoval nezávisle na ostatních. Jeden velký dataset může být rozdělen do stovek až tisícovek takových částí. Potom co se jednotlivé části zpracují, jsou opět sloučeny do finální podoby. Výhodou je také snadná škálovatelnost, kdy při potřebě většího výkonu stačí pouze dokoupit výpočetní kapacitu (uzly) podle potřeby [15].

Zmíněné uzly jsou dvojího typu:

- Řídící uzel – Jak již z názvu vyplývá, řídící uzel má na starost efektivní přidělování a koordinaci úkolů výpočetním uzlům. Také obstarává komunikaci s připojenými aplikacemi.
- Výpočetní uzly – Úlohou výpočetních uzlů je provádět výpočetní operace přidělené řídícím uzlem [15].

Jako výhody jsou uváděny rychlost a efektivita s jakou technologie pracuje oproti podobným technologiím. Dalšími klady je podpora dotazovacích jazyků z rodiny SQL a

snadné nasazení spolu s údržbou. Možnou slabinou může být práce s nestrukturovanými daty, kdy je potřeba předběžné zpracování [15].



Obrázek č. 5: Grafické znázornění Massively Parallel Processing na platformě Microsoft Azure.
(Zdroj: 16)

1.10 Android

Android je operační systém dostupný původně na mobilních zařízeních, ale dnes je možné se s ním setkat také v „chytrých“ multimediálních zařízeních jako je televize nebo multimediální centrum. Jádro systému je založeno na Linuxu, což má za následek, že licence na tento systém podléhají společnostem a organizacím Apache, MIT a BSD, které dovolují volné šíření tohoto operačního systému [18], [20].

Jedná se o nejrozšířenější mobilní systém, a to zejména ze zmíněné možnosti volného šíření a otevřenosti celé platformy. Proto je volen mnohými výrobci mobilních zařízení, kteří upravují dle vlastních funkčních nebo vizuálních požadavků [20].

Vlastníkem celé platformy Android je společnost Google, která jej spolu se skupinou Open Handset Alliance, do níž spadadá 84 technologických společnpostí, spravuje [18], [20].

Tabulka č. 2:Přehled verzí operačního systému Android.
(Zdroj: 19)

Verze	Název	API úroveň
4.0–4.0.2	Ice Cream Sandwich	14
4.0.3–4.0.4	Ice Cream Sandwich	15
4.1	Jelly Bean	16
4.2	Jelly Bean	17
4.3	Jelly Bean	18
4.4	KitKat	19
4.4	KitKat (podpora zařízení typu weawearable)	20
5.0	Lollipop	21
5.1	Lollipop	22
6.0	Marshmallow	23
7.0	Nougat	24
7.1	Nougat	25
8.0.0	Oreo	26
8.0.1	Oreo	27

První verze systému Android (1.0) vyšla před deseti lety, v roce 2008 v USA ve spolupráci s komunikační společností T-Mobile. Produkt měl název T-Mobile G1 a vyrobila jej společnost HTC [20].

Následně nastaly další důležité milníky v roce 2010 a 2011. První zmiňovaný rok 2010 znamenal pro Android dosažení pozice druhého nejprodávanějšího mobilního operačního systému. Ve druhý zmiňovaný rok 2011 se Android dostal na pozici světové jedničky v počtu prodaných zařízení s tímto operačním systémem [20].

Jako vývojový nástroj pro platformu Android slouží Android Studio. Jako nativní jazyk je považována Java, ovšem v roce 2011 byl vydán jazyk Kotlin, který někteří vývojáři preferují na úkor původní Javy. Oficiálně jsou podporovány obě varianty. Kromě

těchto vývojových nástrojů je možné také použít některé z alternativních nástrojů pro multiplatformní vývoj jako je například Xamarin od společnosti Microsoft, který využívá jazyk C# [19], [21].

Distribuce aplikací je dostupná přes Google Play, což je virtuální ochod pro stahování, případně nakupování aplikací, knih a filmů. Možnosti a šíře nabídky obchodu je daná regionálním umístěním uživatele [19].

1.11 Apple iOS

Apple iOS (zkráceně iOS) je proprietární operační systém od společnosti Apple. Jelikož se jedná o uzavřený operační systém, Apple jej distribuuje jen v rámci vlastních zařízení iPhone (mobilní telefon), iPad (tablet) a iPod (mobilní multimediální zařízení) [23].

Historie iOS sahá až do roku 2007, kdy byl představen první iPhone, v té době ještě se systémem iPhone OS, který byl následně spolu s vydáním první verze iPadu v roce 2010, přejmenován na Apple iOS. Mezi mobilními operačními systémy má druhý největší podíl, před je pouze Android [23].

Na rozdíl od Androidu nepoužívá Apple u svých iOS verzí žádná kódová označení, přičemž nové verze operačního systému jsou vydávány s roční pravidelností [23].

Tabulka č. 3: Přehled verzí operačního systému Apple iOS.
(Zdroj: Vlastní zpracování dle: 23)

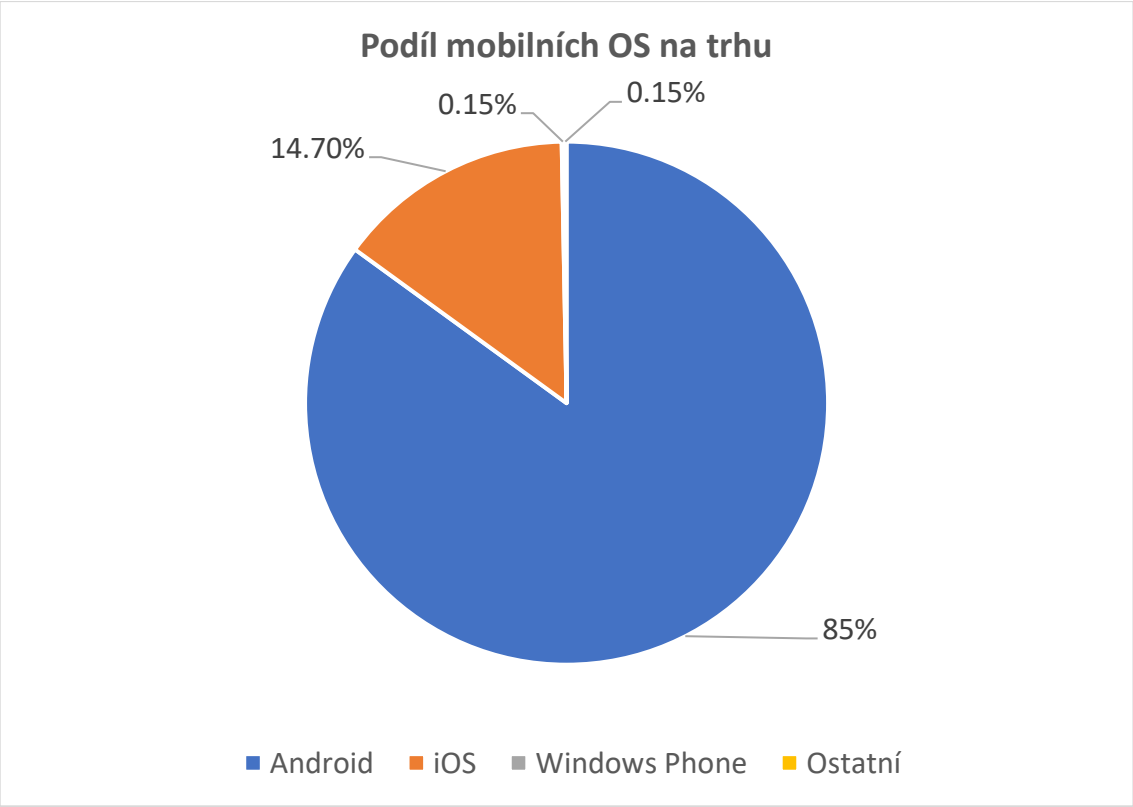
Verze	Rok vydání
iPhone OS 1.x	2007
iPhone OS 2.x	2008
iPhone OS 3.x	2009
iOS 4.x	2010
iOS 5.x	2011
iOS 6.x	2012
iOS 7.x	2013
iOS 8.x	2014
iOS 9.x	2015
iOS 10.x	2016
iOS 11.x	2017

Nativním vývojovým prostředím pro platformu iOS je Xcode, což je vývojový nástroj vyvíjený společností Apple. Pro tvorbu aplikací na tuto platformu je možné využít programovací jazyk Swift nebo starší Objective-C [22].

Apple pro distribuci aplikací a multimediálního obsahu využívá podobný koncept jako Google u Androidu. V případě Applu se však jedná o virtuální obchod s názvem App Store [22].

1.12 Podíl mobilních OS

V předchozích dvou podkapitolách bylo zmíněno pořadí operačních systému na světovém trhu. Tyto údaje vycházejí z údajů společnosti IDC za první kvartál loňského roku (2017). Android a iOS prakticky nemají konkurenci a jsou jedinými systémy na trhu [24].



Graf č. 1: Podíl mobilních operačních systémů na světovém trhu.
(Zdroj: Vlastní zpracování dle: 24)

1.13 Python

Python je vyšší programovací jazyk, jenž byl poprvé vypuštěn v roce 1991. Jedná se o multiplatformní open source[1] jazyk, který podporuje čtyři programátorská paradigmata (funkcionální, imperativní, objektově orientované a procedurální). Python je vyzdvihován především za svou jednoduchost a komplexnost [25].

Má nezměrné pole využití a díky velkému počtu volně dostupných knihoven je hojně využíván v různých odvětvích jako jsou databáze, kde umožňuje práci s běžně užívanými relačními databázemi, ale i tzv. non-SQL databázemi. Díky knihovnám jako NumPy, Pandas nebo SciPY je velice oblíben také v oblasti, kde se pracuje s daty a jejich vizualizacemi. Jako poslední příklad může sloužit také vývoj internetových aplikací, kde může být využit na straně klienta nebo serveru [25].

[1] Open source je označení pro software nebo produkt s volně dostupným kódem. Kdokoli má tedy možnost jej upravovat a volně šířit [25].

Oblíbenost a schopnosti programovacího jazyka Python dokládají také velké společnosti a úřady jako Google, Dropbox, NSA, které jej využívají při své práci a vývoji svých produktů [25].

Python se řídí následující koncepruální hierarchií:

- programy jsou složeny z modulů,
- moduly obsahují tvrzení,
- tvrzení obsahují výrazy,
- výrazy vytváří a zpracovávají objekty [25].

1.14 Hashovací funkce

Hashovací funkce je označení pro jednosměrné funkce s přesně definovanými podmínkami. Jejich úkolem je mapovat řetězce libovolné délky na řetězec konstantní délky, čímž vytvoří tzv. otisk vstupního řetězce, který je také někdy označován jako výtah, hash nebo fingerprint (otisk prstu). Hash je závislý na všech bitech vstupního řetězce [26].

1.14.1 SHA algoritmus

Jde o hashovací funkci navrženou NSA (Národní bezpečnostní agentura USA), kterou následně vydal NIST (Národní institut pro standardy v USA). Dostupných je pět druhů (SHA-1, SHA-224, SHA-256, SHA-384 a SHA-512). Všechny varianty mimo SHA-1 jsou označovány jako SHA-2 a jejich výstupní delky jsou shodné s jejich číselným označením (tzn. SHA-224 má výstupní řetězec o délce 224 znaků apod.) [26]

1.15 API

API (Application Programming Interface – Aplikační programové rozhraní) je sada protokolů a nástrojů pro softwarové aplikace. Aplikace se pomocí něj připojuje k serveru, kde na základě zasílaných požadavků dostává odpovědi [27] .

1.16 Business Intelligence

Tento pojem byl poprvé definován v roce 1989 Howardem Dresnerem následovně: "Business Intelligence je množina konceptů a metodik, které zlepšují rozhodovací proces za použití metrik, nebo systémů založených na metrikách. Účelem procesu je konvertovat velké objemy dat na poznatky, které jsou potřebné pro koncové uživatele. Tyto poznatky můžeme potom efektivně použít například v procesu rozhodování a mohou tvořit velmi významnou konkurenční výhodu." [1, s. 58]

1.17 Diagram toku dat

Diagram toku dat (anglicky DFD – Data Flow Diagram) je jeden z nejpoužívanějších nástrojů pro účely funkčního modelování. Umožňují znázornit návaznosti jednotlivých činností, identifikovat datové vstupy a výstupy, a určit, kdo jednotlivé činnosti provádí. Při tvorbě je možné je vytvářet na různých rozlišovacích úrovních [7].

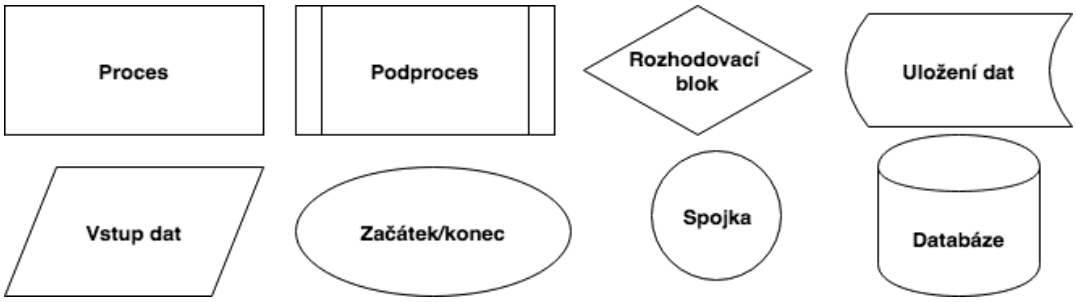
V DFD diagramech se využívají následující prvky:

- Proces – je činností, která transformuje vstupní data na data výstupní.
- Externí entita – je objekt, jenž se nachází v okolí systému, a s kterým proces komunikuje.
- Uložení dat, datová paměť – může mít podobu datového souboru, dokladu nebo sestavy. Má pasivní povahu a slouží pro uložení dat, která se budou zpracovávat následně.
- Pojmenovaný datový tok – reprezentuje přesun dat mezi jednotlivými částmi systému [7].

Obrázek č. 6: Schématické symboly používané v DFD diagramu.
(Zdroj: Vlastní zpracování dle: 7)

1.18 Vývojový diagram

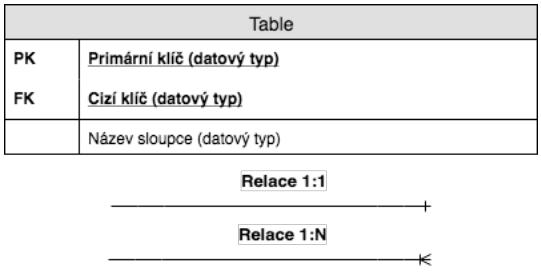
Stejně jako DFD hojně využívaný nástroj pro funkční modelování. Na rozdíl od DFD diagramu se zaměřuje na větvení událostí v závislosti na splnění nebo nesplnění stanovených podmínek [7].



Obrázek č. 7: Schématické symboly používané ve vývojovém diagramu.
(Zdroj: Vlastní zpracování dle: 7)

1.19 Entito-relační diagram

Entito-relační diagram (ERD – Entity-relation diagram) slouží ke grafickému zobrazení tabulek v databázi. Z jeho zobrazení pak lze vyčíst údaje o názvech sloupců, jejich použité klíče a datové typy. Současně ERD diagram umožňuje vizuálně prezentovat relace mezi jednotlivými tabulkami v databázi [7].



Obrázek č. 8: Schématické symboly používané v entito-relačním diagramu.
(Zdroj: Vlastní zpracování dle: 7)

1.20 SWOT analýza

Patří mezi jeden z nejvíce využívaných nástrojů analýzy. SWOT analýza má za úkol posoudit faktory ze čtyř odlišných pohledů:

- silné stránky (Stranghts) - interní faktor,
- slabé stránky (Weaknesses) - interní faktor,

- příležitosti (Opportunites) - externí faktor,
- hrozby (Threats) - externí faktor [28].

Mezi interní faktory jsou takové faktory, jenž může subjekt sám ovlivňovat, naopak externí nemá možnost, jakkoliv ovlivnit a lze se pouze přizpůsobit [28].

2 Analýza současného stavu

V této kapitole bude představené současné řešení využívané společností. Následně bude proveden průzkum trhu s alternativními produkty a celkové zhodnocení analýzy.

2.1 Představení společnosti

Společnost XYZ se zabývá prodejem produktů nabízených přes webový portál a v mobilní aplikaci na platformách Android a Apple iOS.

2.2 Současné řešení

V současné době je ke sběru dat z mobilní aplikace využíván cloudový marketingový nástroj Exponea od stejnojmenné slovenské společnosti. K integraci událostí z aplikací se využívá SDK, které je dostupné pro obě mobilní platformy (Android a iOS) [29].

Exponea nabízí v současném stavu tyto nástroje:

- Kampaně – Jedná se především o skupinu marketingových nástrojů.
  - Scénáře – Slouží k vykonání zvolené akce (například zaslání emailu nebo push notifikace, SMS) na základě stanovených podmínek.
  - Web layers – Jedná se o druh kampaně, která je zobrazována na webové stránce. Ke své funkci využívají HTML/CSS/Javascript. Z důvodu použitých technologií nelze tento nástroj aplikovat na mobilní aplikace.
  - Emailové kampaně – Slouží k zasílání emailových zpráv uživateli na základě časového údaje nebo akce. Podobná funkce jako Scénáře, avšak zaměřená pouze na emaily.
  - Dotazníky – Slouží k vytváření dotazníků a shromažďování odpovědí [29].
- Analytické nástroje – Obsahuje nástroje pro tvorbu analýz a vizualizaci dat.
  - Trychtýř – Jedná se o analýzu, která se využívá k určení konverzí mezi dvěma nebo více událostmi. Metrika, s kterou pracuje, je počet uživatelů. Výsledky se dají dále dělit podle vybraného atributu události.
  - Report – Tento nástroj je velice univerzální a umožňuje graficky (sloupcové grafy, koláčové grafy, spojnicové grafy apod.) nebo tabulkově interpretovat

výsledky zadaných dotazů. Základem jsou metriky, jež můžou být události (v souvislosti s jednou z agregačních funkcí), uživatelé nebo složené formule. Osy jsou následně tvořeny atributy vybraných metrik. Síla reportu spočívá v kombinaci s filtry, které mohou být vázány na uživatele nebo událost.

- Trendy – Jejich úkolem je zobrazování vývoje sledovaných událostí v čase.
- Retence – Slouží k zobrazení jak často a v jakých časových intervalech uživatel vykonává danou událost.
- Segmentace – V Exponei se rozlišují dva druhy segmentací:
  - Zákaznická – Rozlišuje se na úrovní zákazníků.
  - Událostní – Slouží pro segmentaci na úrovní samotných událostí.
- Toky - Zobrazují do větvení událostí ze zvolené výchozí události nebo naopak, které události přecházeli zvolené cílové události.
- Geo analýza – Za pomoci mapových podkladů zobrazuje výskyt uživatelů [29].

- Správa dat – Umožňuje tvořit kritéria pro analýzy a také spravovat události.
  - Správa událostí – Obsahuje seznam všech událostí a jejich stav. Nástroj také umožňuje definování nových nebo úpravu starých událostí, případně jejich třídění a správu skupin. Události se mohou zachytávat automaticky (každá neznámá událost se přiřadí do seznamu událostí) nebo manuálně (zachytávají se pouze události explicitně určené v seznamu).
  - Agregátory – Vytváření kalkulací na základě agregačních funkcí na základě každého zákazníka.
  - Running agregátory – Mají podobnou funkčnost jako klasické agregátory avšak místo zákazníků se vztahují na události.
  - Výrazy – Využívají se k tvoření výrazů, které mohou být založeny na uživatelích nebo událostech (například časový rozestup mezi prvním a posledním nákupem) [29].

Exponea umožňuje také integraci služeb třetích stran. Z databázových nástrojů jsou prozatím dostupné Google BigQuery, Microsoft SQL, MySQL, Oracle a PostgreSQL. Dalšími podporovanými službami jsou například Google AdWords a Google Analytics, Facebook Ads nebo Zapier [29].

Tabulka č. 4: Vyhodnocení výhod a nevýhod Exponei.
(Zdroj: Vlastní zpracování)

Výhody	Nevýhody
Rychlá implementace	Jen webové UI
Real-time	Relativně pomalé (u složitějších požadavků)
Dostupné pro všechny platformy (podporující vybrané prohlížeče)	Žádný přístup ke zdrojovým datům
Jednoduché na ovládání (bez nutnosti psát dotazy)	Omezeno na 256 událostí
Marketingové nástroje	Analýzy omezeny na dostupné nástroje
Automatické zachytávání nových událostí a atributů	Chybí pokročilejší nástroje (data mining, machine learning)
Komplexní řešení	Žádná možnost zásahu do ETL procesu
GDPR certifikace	

2.3 Dostupné služby pro skladování dat

Tato podkapitola se bude zabývat dostupnými řešeními, které jsou v současnosti dostupné na trhu, jenž budou představeny a porovnány.

2.3.1 Redshift

Redshift je řešení datového skladu, jenž nabízí Amazon v rámci své nabídky produktů Amazon Web Services (AWS). Datový sklad je založen na open source objektově-relačním databázovém systému PostgreSQL, který je však na rozdíl od standardní verze obsahuje některé specifické úpravy, zbytek je pak shodný se standardem PostgreSQL. To znamená, že pro práci s datovým skladem Redshift je proto možno vyžívat dotazovací jazyk SQL [30].

Základním prvkem celé architektury je tzv. klastr, který je tvořen jedním nebo více výpočetními uzly. Celá tato struktura je završena vedoucím uzlem, který koordinuje

jednotlivé výpočetní uzly a zároveň obstarává externí komunikaci s klientskou aplikací. Amazon u svého produktu Redshift využívá technologii MPP [30].

Celý klastr může obsahovat jednu nebo více databází, přičemž data jsou uložena v samotných výpočetních uzlech. Jelikož je Redshift relačnš orientovaný systém řízení báze dat, podporuje také stejnou funkcionalitu, a to včetně OLTP funkcí, současně je však optimalizován pro náročné datové analýzy a reportování z obrovských datových sad [30].

Výhodou, kterou také AWS u svého produktu nabízí, je síť certifikovaných partnerů, díky čemuž je možné bez obav vybrat kompatibilní službu zajišťující ETL proces nebo naopak Bussiness Intelligence funkci pro reporting a vizualizaci dat [30].

Nahrávání dat do datového skladu je možné z úložiště Amazon S3, Amazon EMR, vzdálený host s SSH (například Amazon EC2) a Amazon DynamoDB. Přičemž je platná shodná podmínka, která stanovuje maximální velikost jednoho řádku na 4 MB [30].

Redshift nabízí správu na úrovni klastrů a databáze. V obou případech jde o možnost vytváření, správy (monitoring, zálohování, aplikace patchů a upgradů) a mazání. Ovšem na obou úrovních jsou ke správě využívány jiné nástroje:

- Správa klastrů – Správa může být prováděna buď pomocí AWS Management Console, která je přístupná přes webové rozhraní nebo mobilní aplikaci. Další možnosti zahrnují použití kódu skrze Amazon Redshift QUERY API, AWS SDK nebo AWS CLI (příkazová řádka).
- Správa databáze – V tomto případě má uživatel na výběr ze dvou možností. První je klientská aplikace, druhou pak použití SQL příkazů [30].

Amazon nabízí tři možnosti jakými lze datový sklad financovat, přičemž se cena liší podle umístění, výkonu a typu uzlů (Compute – vyšší výkon, ale menší velikost úložiště, Storage – naopak). Uživatel může vybírat ze čtyř aktuálních variant, jež jsou rozepsány v tabulce [30].

Tabulka č. 5: Přehled konfigurací uzlů v Amazon Redshift.
(Zdroj: Vlastní zpracování dle: 30)

Metrika	Compute (dc2)		Storage(ds2)	
Velikost	large	8xlarge	large	8xlarge
vCPU[2]	2	32	4	36
ECU[3]	7	99	14	166
RAM	15 GB	244 GB	31 GB	244 GB
Úložiště/uzel (SSD)	160 GB	2,56 TB	2 TB	16 TB
I/O[4]	0,6 GB/s	7,5 GB/s	0,4 GB/s	3,3 GB/s
Max. kapacita	5,12 TB	326 TB	64 TB	2 PB
Rozsah uzlů	1–32	2-128	1-32	2-128

Zmíněné možnosti financování skladu jsou následující:

- On-demand – Způsob funguje na principu platby za reálně využité zdroje na hodinové bázi.
- Spectrum – U tohoto typu financování se platí za SQL dotazy, které jsou zaokrouhlovány na MB s tím, že minimální hodnota je 10 MB. Zpoplatněny však nejsou dotazy pro práci s tabulkami (CREATE/ALTER/DROP), oddíly a chybné dotazy.
- Reserved Instance – Jde o formu předplatného na období jednoho roku nebo tří let, čímž lze dosáhnout výrazných slev až do 70 % oproti On-demand možnosti. Předplatné zahrnuje také zálohu dat v úložišti S3. Cena se také odvíjí od toho, zda je předplatné uhrazeno najednou na začátku cyklu (All Upfront), část na počátku a zbytek v průběhu smluvního období (Partial Upfront) nebo platby podle využití zdrojů v průběhu období (No upfront) [30].

[2] Počet virtuálních CPU na uzel [30].
[3] Počet výpočetních jednotek Amazon EC2 na uzel [30].
[4] Propustnost [30].

Tabulka č. 6: Cenové sazby u Amazon Redshift.
(Zdroj: Vlastní zpracování dle: 30)

	Délka závazku	dc2.large	dc2.8xlarge	ds2.xlarge	ds2.8xlarge
On-demand	-	$0,30/h	$5,60/h	$0,95/h	$7,60/h
Spectrum	-				$5,00/terabyte dat
Reserved AU	1 rok	$0,1918/h	$3,792/h	$0,613/h	$4,903/h
	3 roky	$0,1126/h	$1,801/h	$0,2656/h	$2,125/h
Reserved PU	1 rok	$0,1956/h	$3,87/h	$0,6254/h	$5,003/h
	3 roky	$0,1204/h	$1,927/h	$0,2842/h	$2,273/h
Reserved NU	1 rok	$0,24/h	$4,40/h	$0,76/h	$6,00/h
	3 roky	-	-	-	-

Tabulka č. 7: Přehled parametrů Amazon Redshift.
(Zdroj: Vlastní zpracování dle: 30)

Parametr	Hodnota
Typ databáze	SQL
Datové typy – datum	DATE, TIMESTAMP, TIMESTAMPTZ
Datové typy - text	CHAR, VARCHAR, BPCHAR, TEXT
Datové typy – čísla s pohyblivou čárkou	DECIMAL, REAL, FLOAT
Datové typy – čísla s pevnou čárkou	SMALLINT, INT, BIGINT
Datové typy - strukturované	-
Datové typy - ostatní	BOOLEAN
Podporované ovladače	ODBC, JDBC
Omezení databáze	100 databází na účet a 9 900 tabulek na uzel
Omezení tabulek	20 000 oddílů/tabulka
Počet sloupců v tabulce	1 600

2.3.2 BigQuery

BigQuery je produkt od společnosti Google a jeho sady produktů Google Cloud Platform (GCP). Datový sklad využívá technologie Dremel, která umožňuje dotazování

nad velkým množstvím dat a byl inspirací také pro open source produkty od Apache, konkrétně Drill a Impala. Ačkoli se jedná o proprietární technologii Googlu, k dotazování lze používat standardní syntaxi jazyka SQL. BigQuery také dokáže komunikovat pomocí REST API [31].

K ukládání dat používá souborový systém Capacitor, který se stará o kompresi dat, operace s metadaty nebo kešování dat. Samotná data jsou pak uložena na platformě Colussus. Celá architektura funguje stejně jako u Redshiftu na principu klastrů [31].

Google, shodně jako Amazon, nabízí pole doporučených partnerů třetích stran, kteří poskytují služby pro nahrávání dat do datového skladu nebo naopak pomoc s jejich analýzou a vizualizací [31].

Nahrávání dat do skladu je možné řešit několika způsoby.

- Nahrávání z databáze Google Cloud Storage, kde se data předávají pomocí textových souborů, Apache AVRO nebo nahráním ze záloh.
- Nahrávání lokálních dat pomocí webového rozhraní nebo CLI (Command-Line Tool – příkazový řádek)
- Proudění dat přímo do datového skladu. Oproti předchozím variantám je výhodou přímé zapisování, kdy nedojde ke zpoždění z důvodu vykonávání procesu nahrávání dat.
- Další možností je jazyk DML (Data Manipulation Language), který umožňuje aktualizovat, vkládat a mazat záznamy z tabulek. Platí tato omezení:
- UPDATE/DELETE – maximálně 96 údajů v tabulce/den nebo 10 000 údajů v daném projektu/den
- INSERT – maximálně 1 000 údajů v tabulce/den
- Dataflow je jeden z produktů patřících do portfolia GCP. Data je možné vkládat v reálném čase (streamováním) nebo dávkami, což je využíváno u historických dat [31].

BigQuery má množinu podporovaných datových typů. Nejoptimističtější varianta, kdy se shodují podporované datové typy na vstupu i výstupu, jen překonvertuje vstup do daného datového typu a zapíše jej do datového skladu. Pokud datový typ na vstupu není podporován, je zkonvertován do řetězce a zapsán. Na závěr může nastat situace, kdy z technických příčin není možné vstupní typ poznat. V takovém případě je zápis s hodnotou

null. Na rozdíl od Redshiftu je u BigQuery možné spravovat datasety, kde jsou nastavené limitace v podobě nemožnosti aplikace patchů nebo update příkazů [31].

Tabulka č. 8: Cenová sazba v Google BigQuery.
(Zdroj: Vlastní zpracování dle: 31)

	Úložiště	Dlouhodobé úložiště	Cena za přenos	Dotazy
Cena	$0,02/GB/měsíc	$0,01/GB/měsíc	$0,05/GB	$5/TB

Dlouhodobé úložiště je tabulka, u které se minimálně 90 dní neprovedly žádné editace. Ostatní úkony jako import/export/kopírování dat a operace s metadaty jsou zdarma. Přičemž je každý měsíc zdarma 10 GB úložiště a první TB dotazů [31].

Tabulka č. 9: Přehled parametrů v Google BigQuery.
(Zdroj: Vlastní zpracování dle: 31)

Parametr	Hodnota
Typ databáze	SQL
Datové typy – datum	DATETIME, TIMESTAMP
Datové typy - text	STRING
Datové typy – čísla s pohyblivou čárkou	FLOAT
Datové typy – čísla s pevnou čárkou	INTEGER
Datové typy - strukturované	-
Datové typy - ostatní	BOOLEAN, BYTES, RECORD
Podporované ovladače	ODBC, JDBC
Omezení databáze	-
Omezení tabulek	-
Počet sloupců v tabulce	10 000

2.3.3 Azure SQL Data Warehouse

Azure SQL Data Warehouse je produkt od společnosti Microsoft spadající do portfolia cloudových nástrojů Azure. Hlavní předností jsou technologie Massively Parallel Processing (MPP), která umožňuje rychle poskytovat odpovědi na požadované dotazy napříč petabyty dat, a PolyBase, jenž může provádět dotazy pomocí T-SQL nad

daty z externích zdrojů (Hadoop, Azure Blob Storage). Stejně jako v ostatních databázových produktech i zde Microsoft využívá svou verzi jazyka SQL – Transaction SQL (T-SQL) [32].

Architektura opět připomíná předchozí řešení. Na nejnižší úrovni je samotné úložiště, které má v tomto případě název Azure Storage. Data v úložišti jsou rozložena mezi jednotlivé distribuce (základní jednotka skladiště a zpracování paralelních dotazů) pro optimální výkon, přičemž lze zvolit mezi hash-rozdělením tabulek (pro spojovací a agregační operace nad velkými tabulkami), Round-robin rozdělení tabulek (pro jednoduché nahrávací tabulky) a replikované tabulky (pro malé tabulky). Nad samotným úložištěm jsou výpočetní uzly, které jsou sjednoceny v řídícím uzlu, který obstarává komunikaci s aplikacemi a ke své činnosti využívá právě technologii MPP. Pro určování potřebného výkonu se pak používají tzv. DWU (Data Warehouse Unit) a cDWU (Compute Data Warehouse Unit), což jsou jednotky s normalizovaným výpočetním výkonem [32].

Pro produkt SQL Data Warehouse jsou rovněž doporučení partneři, jimiž jsou společnosti třetích stran s produkty pro integraci dat do skladu a také jejich prezentaci a vizualizaci (BI) [32].

Pro nahrávání dat sám Microsoft doporučuje využít PolyBase, kde uživatel může využít jednu z možností (maximálně 1 MB na řádek):

- Kombinace s T-SQL, kdy jsou data nahrávána z Azure databází Blob storage nebo Data Lake Store.
- Kombinace s SSIS, která slouží, pokud jsou data uložena na SQL Serveru, bez ohledu na to jedná-li se lokální server nebo server běžící v cloudu.
- Kombinace s Azure Data Factory (ADF), pomocí něhož lze informace získat z podporovaných úložišť jako je Redshift, MySQL, Oracle, PostgreSQL a další [32].

V případě, že uživatel se chce mermomocí vyhnout nahrávaní pomocí PolyBase má možnost v podobě bcp (bulk copy program utility – nástroj příkazového řádku pro přesun dat mezi úložišti) nebo SQL Bulk Copy API (třída frameworku .NET pro operaci nahrávání dat mezi datovými zdroji), která neprochází skrz Blob storage, současně však

jde o řešení určené jen pro malé dávky dat s menším výkonem, než je tomu s využitím PolyBase [32].

Správu skladu je možno provádět na úrovni výpočetních jednotek, serveru (každá instance datového skladu musí běžet na SQL Serveru) i datového skladu. První dvě možnosti jsou dostupné přes webový portál, pomocí nástroje PowerShell nebo REST API. Správa datového skladu může probíhat také prostřednictvím webového portálu a za pomoci jazyka T-SQL [32].

Pro běh datového skladu na platformě Azure je potřeba zakoupit zvlášť výpočetní jednotky (cDWH), které jsou optimalizované pro elasticitu (pro častou úpravu výkonu) a výpočetní operace (lepší výkon při nahrávání dat a dotazování), a úložiště. Cena se opět určuje také podle oblasti [32].

Tabulka č. 10: Cenové sazby v Microsoft Azure SQL Data Warehouse podle výpočetního výkonu. (Zdroj: 32)

Úroveň	cDWU[5]	Cena za hodinu
DW1000c	1 000	$6.96/h
DW1500c	1 500	$10.44/h
DW2000c	2 000	$13.92/h
DW2500c	2 500	$17.40/h
DW3000c	3 000	$20.88/h
DW5000c	5 000	$34.80/h
DW6000c	6 000	$41.76/h
DW7500c	7 500	$52.20/h
DW10000c	10 000	$69,60/h
DW15000c	15 000	$104,40/h
DW30000c	30 000	$208,80/h

[5] cDWU (Compute Data Warehouse Unit) je výpočetní jednotka s normalizovaným výpočetním výkonem [32].

Tabulka č. 11: Cenové sazby v Microsoft Azure SQL Data Warehouse podle velikost úložiště.
(Zdroj: 32)

Úroveň	Velikost disku	Cena/měsíc	Počet I/O /disk	Propustnost
P4	32 GB	$5,28	120	25 MB/s
P6	64 GB	$10,21	240	50 MB/s
P10	128 GB	$19,71	500	100 MB/s
P15	256 GB	$38,02	1100	125 MB/s
P20	512 GB	$73,22	2300	150 MB/s

Tabulka č. 12: Přehled parametrů v Microsoft Azure SQL Data Warehouse.
(Zdroj: Vlastní zpracování dle: 32)

Parametr	Hodnota
Typ databáze	SQL
Datové typy – datum	DATETIMEOFFSET, DATETIME(2), SMALLDATETIME, DATE, TIME
Datové typy - text	NVARCHAR, NCHAR, VARCHAR, CHAR, VARBINARY
Datové typy – čísla s pohyblivou čárkou	FLOAT, REAL, DECIMAL, MONEY, SMALLMONEY
Datové typy – čísla s pevnou čárkou	BIGINT, INT, SMALLINT, TINYINT
Datové typy - strukturované	-
Datové typy - ostatní	BIT, VARBINARY, BINARY, UNIQUEIDENTIFIER
Podporované ovladače	ODBC, JDBC
Omezení databáze	Velikost max. 240 TB a max. 2 bilióny tabulek v databázi
Omezení tabulek	-
Počet sloupců v tabulce	1 024

2.3.4 Snowflake

Snowflake je produktem společnosti Snowflake Computing, jenž je středně velká a mladá společnost (založena 2012, produkt na trhu od 2015). Snowflake je datový sklad

využívající k dotazování standardní syntaxí jazyka SQL. Architektura datového skladu je založena na principu klastrů, přičemž je rozdělena do třech úrovní – úložiště, výpočetní úroveň a služby [33].

- Úložiště – Je založeno na databázi od AWS s názvem S3, která je automaticky upravována podle potřeb zákazníka.
- Výpočetní úroveň – Provádí operace jako nahrávání dat, transformace a samotné dotazování. Při těchto operacích uživatel vytvoří "virtuální sklady", které mají schopnost přistupovat do jakékoli databáze z první úrovně (kde má uděleny přístupy). Následně se získaná data uloží do cache paměti.
- Služby – Úkolem nejvyšší úrovně je správa metadat, zabezpečení, přístupů do systému apod. Tato úroveň také zajišťuje komunikaci s klientskými aplikacemi [33].

Nahrávání dat do Snowflake může probíhat hromadně pomocí souborů CSV, TSV apod. nebo souborů JSON, Avro, ORC, Parquet a XML. Takto nahrávané soubory můžou být uloženy na lokálním zařízení nebo v databázi Amazon S3. Druhou možností je nahrávání dat do skladu za pomoci Snowpipe, jež je také službou Snowflake Computing. Snowpipe funguje na principu volání veřejného REST rozhraní, poté se služba sama postará o nahrání dat a jejich uložení do datového skladu. Obdobně jako hromadné nahrávání probíhá také export, který však oproti nahrávání podporuje jen formáty CSV, TSV apod. doplněné o JSON. Zároveň je možný export taktéž do databáze Amazon S3 [33].

Management lze provádět na úrovni virtuálních datových skladů, jenž jsou zdrojem výpočetního výkonu a na úrovni samostatných databází. V obou případech se k vykonávání operací využívá jazyk SQL [33].

Cena za služby Snowflake se odvíjí od jedné z pěti edic, přičemž je vždy možnost bez úvazku (On-demand) nebo s úvazkem (Capacity). Edice se kromě ceny za výpočetní uzly liší také výčtem dostupných služeb. Celkové ceny služeb se mění v závislosti na oblasti [33].

Tabulka č. 13: Přehled tarifů ve službě Snowflake.
(Zdroj: Vlastní zpracování dle: 33)

	Standard Edition	Premier Edition	Enterprise Edition	Enterprise for Sensitive Data	Virtual Private Snowflake
On-Demand - Úložiště	$45/TB/m	$45/TB/m	$45/TB/m	$45/TB/m	$45/TB/m
On-Demand - Výpočetní výkon	$2,7/h	$3/h	$4/h	$5,4/h	$8,1/h
Capacity - Úložiště	$24,5/TB/m	$24,5/TB/m	$24,5/TB/m	$24,5/TB/m	$24,5/TB/m
Capacity - Výpočetní výkon	Individuální	Individuální	Individuální	Individuální	Individuální

Tabulka č. 14: Přehled parametrů ve službě Snowflake.
(Zdroj: Vlastní zpracování dle: 33)

Parametr	Hodnota
Typ databáze	SQL
Datové typy – datum	DATE, DATETIME, TIME, TIMESTAMP, TIMESTAMP (LTZ, NTZ, TZ)
Datové typy - text	CHAR, VARCHAR, TEXT, STRING
Datové typy – čísla s pohyblivou čárkou	FLOAT, DOUBLE, REAL
Datové typy – čísla s pevnou čárkou	DECIMAL, NUMERIC, INT, INTEGER, BIGINT, SMALLINT, TINYINT, BYTEINT
Datové typy - strukturované	VARIANT, ARRAY, OBJECT
Datové typy - ostatní	BINARY, VARBINARY, BOOLEAN
Podporované ovladače	ODBC, JDBC
Omezení databáze	-
Omezení tabulek	-
Počet sloupců v tabulce	-

2.3.5 MongoDB Atlas

Dalším je produkt od společnosti MongoDB s názvem Atlas. Všechny produkty společnosti jsou postaveny na open source databázovém systému jménem MongoDB, jenž je tzv. NoSQL řešením. Na rozdíl od klasických majoritních (relačně orientovaných databázových systémů) databázových systémů je MongoDB založeno na dokumentově orientovaném systému, jenž využívá soubory podobné formátu JSON. Atlas je DaaS (Database as a Service) řešením, přičemž si zákazník může zvolit z nabídky Amazon Web Service, Google Cloud Platform nebo Microsoft Azure [34].

K nahrávání dat jsou využívány textové soubory JSON nebo CSV, přičemž je lze nahrát pomocí čtyř rozdílných programů:

- Live Migrate – Jedná se o nepřetržitou synchronizaci replikovacího setu (replikovací set je modgod (stará se o žádosti, spravuje přístupy k datům a provádí operace na pozadí) instance, která se stará o údržbu stejné sady dat zdrojové MongoDB databáze.
- MoningoMirror – Je utilita, která provádí stejnou funkci jako Live Migrate avšak ke své funkci nepotřebuje vypnout existující replikační set nebo aplikaci.
- Mongostore – Využívá programy mongodump a mongostore k zavedení dat do Atlas klastru z existující samostatné MongoDB, replikačního setu nebo rozděleného klastru.
- Mongoimport – Slouží k naimportování dat ze souborů JSON nebo CSV do MongoDB Atlas [34].

Jelikož je databáze MongoDB dokumentově orientovaná, pro ukládání dokumentů využíván formát BSON (Binary JSON – binární kódovaná serializace dokumentů typu JSON). Od použitého formátu BSON se odráží také nabídka dostupných datových typů [12], [34].

MongoDB Atlas provádí nejrůznější operace s klastry jako jejich vytváření, mazání nebo přesouvání. Současně podporuje správu samotných databází. Tyto operace se provádějí skrze webové rozhraní správy projektu [34].

MongoDB má stejně jako Snowflake několik edicí předplatného – Free, Essential a Proffesional. Dále se cena odráží od poskytovatele cloudu, čímž vzniká mnoho různých kombinací [34].

Tabulka č. 15: Přehled cen a edicí u MongoDB Atlas.
(Zdroj: Vlastní zpracování dle: 34)

	Paměť	Úložiště	Cena
AWS	Sdílená - 256 GB	512 MB - 4 TB	Zdarma - $18,75/h
GCP	Sdílená - 120 GB	2 GB - 4 TB	$0,013/h - $12,38/h
Azure	Sdílená - 140 GB	2 GB - 4TB	$0,013/h - 11,29/h

Tabulka č. 16: Přehled parametrů u MongoDB Atlas.
(Zdroj: Vlastní zpracování dle: 34)

Parametr	Hodnota
Typ databáze	No – SQL
Datové typy – datum	DATE, TIMESTAMP
Datové typy – text	STRING
Datové typy – čísla s pohyblivou čárkou	DOUBLE
Datové typy – čísla s pevnou čárkou	INTEGER
Datové typy – strukturované	OBJECT, ARRAY
Datové typy – ostatní	BINARY DATA, UNDEFINED, OBJECT_ID, BOOLEAN, NULL, SYMBOL, REGULAR EXPRESSION, JAVASCRIPT, MIN_KEY, MAX KEY
Podporované ovladače	ODBC, JDBC
Omezení databáze	Maximální velikost 32 TB
Omezení tabulek	Maximální velkost 16 MB
Počet sloupců v tabulce	-

2.3.6 Drill

Drill je open source projekt od Apache Software Foundation inspirovaný Dremelem od Googlu. Jednou z největších výhod Drillu je schopnost spojovat v rámci jednoho dotazu data z několika různých zdrojů (viz. podporované databáze a systémy) [35].

Jádro Drillu je tzv. Drillbit, který obstarává požadavky od klienta, zpracovává dotazy a nazpět vrací výsledky klientovi. Drillbit může být nainstalován a provozován na uzlech Hadoop klastru. Přestože Drill využívá Hadoop klastr, není s nim nijak spojena a může pracovat v jakémkoli distribuovaném prostředí klastru. Jediným předpokladem pro Drill je Zookeper, jenž udržuje členství klastru a kontroluje stav. Drillbit je složen ze tří modulů:

- RPC endpoint – stará se o komunikaci s klienty.
- SLQ parser – využívá Calcite, což je open source SQL framework pro parsování příchozích dotazů.
- Storage plugin interface – vrstva jenž přistupuje k datovému zdroji. Následně poskytuje systému informace o meta datech, přístup pro čtení a zápis, informace o lokaci dat a množinu pravidel pro efektivní práci s daty pro specifický zdroj dat [35].

Data v Drillu nemají formu relační databáze, ale jsou reprezentovány pomocí tzv. JSON datového modelu, který je podobný modelu databáze MongoDB [35].

Drill podporuje mnoho datových zdrojů, čehož lze docílit pomocí pluginu, který vytváří prostředníka mezi datovým zdrojem a Drillem. Pluginy mohou být podle požadavků přepisovány, tak aby plně vyhovovaly požadavkům uživatele, přičemž správa je možná pomocí čtyř nástrojů:

- Drill shell – příkazová řádka, s množinou příkazů pro správu Drillu.
- Drill Web Console – správa prostřednictvím webového rozhraní v prohlížeči.
- ODBC/JDBC – vykonávání příkazů přes software podporující zmíněné ovladače.
- C++ API – rozhraní pro vykonávání pomocí jazyka C++ [35].

Pro nahrání dat do Drillu je možné využít Avro, textové soubory (CSV, TSV a PSV), JSON, Parquet (výchozí), MapR-DB a Hadoop Sequence Files. Přičemž jako zdroj dat můžou sloužit nejrůznější databáze a souborové systémy jako Amazon S3, Windows

Azure Blob Storage, Google Cloud Storage, MongoDB, HBase, Hive nebo Hadoop. Nad různými zdroji (i těmi patřící do skupiny NoSQL databází) umožňuje provádět dotazování formou standardního SQL [35].

Drill má také REST API, jenž dovoluje jeho kombinaci s klasickými BI nástroji jako Tableau, Qlik, Excel, SAS a dalšími [35].

Tabulka č. 17: Přehled parametrů Drillu.
(Zdroj: Vlastní zpracování dle: 35)

Parametr	Hodnota
Typ databáze	No – SQL
Datové typy – datum	INTERVAL, TIMESTAMP (JDBC), DATE
Datové typy – text	CHAR, VARCHAR
Datové typy – čísla s pohyblivou čárkou	DECIMAL, FLOAT, DOUBLE
Datové typy – čísla s pevnou čárkou	BIGINT, INTEGER, SMALLINT
Datové typy – strukturované	-
Datové typy – ostatní	BOOLEAN, VARBINARY
Podporované ovladače	ODBC, JDBC
Omezení databáze	-
Omezení tabulek	-
Počet sloupců v tabulce	-

2.3.7 Impala

Stejně jako u Drillu se jedná o open source MMP databázový engine od Apache Software Foundation běžící na Apache Hadoop. Opět se jedná o produkt inspirovaný Google, konkrétně jeho databází Google F1. K vytváření dotazů využívá standardní SQL [36].

Architektura Impaly je rozdělena do tří komponent:

- Deamon – běží na každém datovém uzlu klastru. Stará se o čtení a zápis do datových souborů, přijímá dotazy a následně se stará o jejich distribuci v klastru a výsledky opětovně vrací centrálnímu uzlu. Deamon je neustále v kontaktu s

statestore (viz. Následující odrážka). Současně dostává zprávy o vytvoření klastru, úpravě, smazání jakéhokoli typu objektu.

- Salestore – sleduje "zdraví" všech datových uzlů v klastru a své výsledky sděluje Deamonu.
- Cataloge Service – stará se o přenos změn z SQL příkazů do všech datových uzlů [36].

Správa pomocí Impala Shell je příkazová řádka pro vykonávání operací nad databází, k provádění úkonů je možné také používat soubory se skripty. Impala dokáže pracovat také s Amazon S3 a Azure Data Lake Store [36].

O distribuci se stará společnost Cloudera jenž využívá Impalu ve svém produktu Analytic Database. Databáze je postavena na Amazon S3. Roční cena je buďto $8000 při předplacení nebo hodinová sazba $0,24 [36].

Tabulka č. 18: Přehled parametrů Impaly.
(Zdroj: Vlastní zpracování dle: 36)

Parametr	Hodnota
Typ databáze	No – SQL
Datové typy – datum	TIMESTAMP
Datové typy – text	STRING, CHAR, VARCHAR
Datové typy – čísla s pohyblivou čárkou	DECIMAL, DOUBLE, FLOAT, REAL
Datové typy – čísla s pevnou čárkou	BIGINT, INTEGER, SMALLINT, TINYINT
Datové typy – strukturované	ARRAY, STRUCT, MAP
Datové typy – ostatní	BOOLEAN
Podporované ovladače	ODBC, JDBC
Omezení databáze	-
Omezení tabulek	-
Počet sloupců v tabulce	-

2.4 Integrační ETL/služby

Tato podkapitola bude obsahovat přehled integračních nástrojů pro sběr dat v aplikaci a jejich následný přenos do datového skladu.

2.4.1 Alooma

Alooma je nástroj pro integraci dat do vybraných datových skladů v reálném čase. Základním prvkem celého systému jsou události (eventy). Událost je každý prvek, jenž se dostane na SDK služby (řádek z databáze, textového souboru apod.), takže například každý řádek ze zdrojového souboru je brán jako jednotlivá událost. Následně je tato událost převedena do JSON struktury. Druhým podstatným prvkem je event type (typ události), což je interní datový konstruktor, jenž je vytvářen automaticky pro reprezentaci datové struktury určitého typu události. Pro každý vstup je vytvořen jeden a více typů událostí [37].

Pro přenos dat je využíváno dvou základních nástrojů:

- Mapper – Definuje, které části událostí jsou přebírány a replikovány do cílového skladu. Nástroj přizpůsobuje výstup podle toho, do jakého skladu jsou data nahrávána.
- Code Engine – Určuje, jak jsou události na vstupu zpracovávány – transformace, spojování, doplnění apod. Tyto akce probíhají v reálném čase a celý nástroj je založen na Pythonu.
- Restream Queue – Úkolem nástroje je zabránit ztrátě dat. Takový případ může nastat například při převodu dat do nevhodného datového formátu. Data, která při nahrávání neprošla z důvodu chyby jsou uložena a připravena k opravě.
- S3 Retention – Jedná se o zálohu dat, kdy jsou tzv. raw data (původní podoba vstupu) uložena do databáze Amazon S3 [37].

Jako výstup slouží nejrůznější datové sklady a technologie. Z těch, jenž byly zmíněny v předchozí podkapitole je dostupná podpora pro Amazon Redshift, Google BigQuery, Azure SQL Data Warehouse a Snowflake, naopak MangoDB je podporováno jen na vstupu. Naopak jako vstup můžou sloužit databáze Oracle, SQL Servery, marketingové nástroje (Adwords, Facebook Ads, Google Analytics) a především data

mobilních aplikací, případně jakékoliv data posílána skrze REST API. Cena je daná v závislosti na množství přenesených a zpracovaných událostí [37].

2.4.2 Segment

Segment slouží ke sběru dat nebo jejich replikaci ze zdrojového systému do podporovaného datového skladu. Zachytávání událostí probíhá pomocí sady vlastních knihoven, které jsou navrženy pro různé zdroje. API rozhraní poskytuje možnosti šesti druhů volání:

- Identify – Základní identifikace uživatele. Obsahuje unikátní ID uživatele, které se dělí na Anonymous ID, pokud je uživatel neznámý a User ID pokud je uživatel rozeznán na základě již nasbíraných dat v databázi.
- Track – Popis události, kterou uživatel vykonal, doplněná o dodatečné informace.
- Page/Screen – Druh volání se rozlišuje podle zařízení (Page je určen pro webové stránky, zatímco Screen je určen pro mobilní aplikace).
- Group – Užívá se pokud je identifikovaný uživatel svázán s nějakou skupinou nebo společností.
- Alias – Je určen pro sjednocení dvou uživatelských identit do jedné.

Nahrávání dat do cílových úložišť může probíhat dvěma způsoby:

- Retries – Knihovny od Segmentu zaručují dodání dat navzdory problémům s připojením nebo problémům zařízení. Na mobilním zařízení jsou zaznamenané události odesílány na pozadí, kde se ukládají do fronty a zapisují na disk. Pokud se sadu nasbíraných událostí nepodaří odeslat napoprvé, odesílání se opakuje, dokud se úspěšně nezdaří. Podobný princip funguje také při odesílání ze Segmentu do cílového úložiště, kdy v případě selhání je akce opakována až pětkrát v následující půl hodině.
- Replays – Data jsou nahrány ze zálohy Segmentu, jenž je uložena v databázi Amazon S3. Segment v záloze uchovává data po dobu šesti měsíců [38].

Data se po odeslání uchovávají ve zvoleném datovém skladu, který má schéma vytvořené Segmentem. Schéma obsahuje tabulky odpovídající API voláním (Aliases, Groups, Indentifies, Screent, Pages, Tracks), navíc doplněné o Accounts, Users a tabulky

pro vlastní události. Podporované datové typy jsou Timestamp, Integer, Float, Boolean a Varchar [38].

Segment podporuje zachytávání událostí jak na webu, tak i mobilních aplikacích, pomocí HTTP nebo Pixel trackingu (v místech, kde není možné spustit kód). Kromě toho také může převádět data mezi cloudovými aplikacemi třetích stran do určeného datového skladu. Z představených datových skladů jsou k dispozici jen Amazon Redshift a Google BigQuery. Obrovskou výhodou Segmentu je z pohledu firmy také možnost nahrávání dat do Exponei, která je využívána jako současné řešení [38].

2.5 SWOT analýza

V této podkapitole budou pomocí SWOT analýzy určeny faktory implementace nového systému.

Tabulka č. 19: SWOT analýza.
(Zdroj: Vlastní zpracování)

Silné stránky	**Slabé stránky**
Různé způsoby získávání dat (SQL, BI)	Nutnost údržby datového skladu
Využití data miningu	
Přístup k datům skrze API	
Možnost úpravy dat v datovém skladu	
Příležitosti	**Hrozby**
Možnost volby služeb na jednotlivých úrovních	Problémy v komunikaci v důsledku dvou rozdílných služeb
Podpora machine learning	Nesplnění výkonových očekávání
Absence omezení dané vývojáři systému	
Velikost datového skladu omezena pouze technologickými limity	

2.6 Zhodnocení analýzy

Z provedené analýzy vyplývá, že současně používané řešení v podobě produktu Exponea je uživatelsky přívětivé a komplexní řešení, jenž pokrývá procesy od

zachytávání událostí v aplikaci a webu, přes jejich nahrávání do systému, až po vizualizaci a další operace založené na datech. Výhodou je také uživatelsky přívětivé rozhraní, včetně vytváření dotazů.

Ovšem v aktuální situaci požadavky společnosti narážejí na limity produktu v počtu možných událostí, ale také v rychlosti a schopnosti Exponei při práci s velkými datovými soubory.

Z důvodu nedostatků Exponei byly srovnány aktuálně dostupná řešení v oblasti datových skladů a ETL služeb. Na rozdíl od Exponei nejde o komplexní řešení, ale celý proces bude složen z několika služeb, což zaručí širší možnosti pro práci s daty v jednotlivých fázích a zároveň větší kontrolu. Například bude možné sjednotit hodnoty atributů z různých platforem, které v Exponei vystupují jako rozdílné hodnoty (klasicky datový typ Boolean, kdy jsou hodnoty True/False a 1/0). Avšak hlavní motivací je vybudování vlastního datového skladu, kde bude mít společnost plný přístup a kontrolu nad svými daty, platforma zajistí rychlejší dotazování nad velkými objemy dat a možnost využití pokročilých nástrojů jako je data mining a machine learning, které mohou zlepšit personalizaci nabídky.

3 Vlastní návrh řešení

Obsahem této kapitoly bude volba ETL služby a datového skladu z předchozí kapitoly. Následně proběhne implementace tohoto systému do aplikace a odzkoušení jeho funkcionality. Na závěr kapitoly bude provedeno ekonomické zhodnocení celého projektu.

3.1 Výběr služby

V analýze současného stavu bylo vybráno sedm služeb pro datový sklad a dvě pro zajištění ETL procesu. Na základě vlastností jednotlivých produktů a konzultaci s ostatními týmy ve společnosti byly zvoleny tyto řešení:

- ETL služba – pro službu zajišťující ETL proces byla zvolena Alooma. Důvodem pro volbu právě této služby je možnost integrace všech důležitých zdrojů (mobilní aplikace a do budoucna také webové aplikace). Dále poskytuje programovou úpravu zasílaných událostí v reálném čase v jazyku Python (podporuje také integraci verzovací služby Git) a současně zálohu zasílaných dat do úložiště Amazon S3 v původní podobě.
- Datový sklad – Jako datový sklad byl zvolen Redshift od AWS (Amazonu). Důvodem volby tohoto produktu byl fakt, že firma již používá jiné služby platformy AWS. Současně se jedná o zavedeného poskytovatele s vyspělou infrastrukturou a zákaznickou podporou, čímž společnosti jako cílovému zákazníkovi odpadne starost o dostupnost a zálohu informací. Transakční databáze společnosti jsou postaveny na databázovém systému PostgreSQL, jehož syntaxe je použita pro dotazování také v Redshiftu v důsledku čehož odpadne nutnost měnit stávající návyky uživatelů.

3.2 Implementace datového skladu

Před samotnou implementací je nutné vytvořit datový sklad. Vytváření skladu probíhá ve webovém rozhraní AWS. Jako první krok je nutné zvolit jméno klastru, databáze a administrátorské údaje.

Jako další krok následuje volba typu uzlů. Vzhledem k současným podmínkám je plně dostačující typ dc2.large. Compute uzlu byla dána přednost oproti Storage uzlu hlavně z důvodu vyššího výpočetního výkonu, jelikož dostupné úložiště, při zvolené verzi, je pro zamýšlený záměr plně dostačující, a to i s ohledem na rostoucí uživatelskou základnu, případně integraci dalších služeb a platforem.

Přes vše zmíněné v předchozím odstavci je vhodné zvolit typ klastru Multi Node, což zajistí případnou škálovatelnost datového skladu v případě potřeby většího výkonu nebo úložiště.

Následně je zvolena VPC skupina, v rámci, které jsou zaneseny výjimky přístupových IP adres nezbytných pro komunikaci ETL služby s datovým skladem a přístup SQL klientů vybraných uživatelů.

Na závěr je z důvodu bezpečnosti v datovém skladu vytvořena identita, jež povolí ETL službě zapisovat data do datového skladu.

Pro napojení datového skladu na ETL službu Alooma, je nezbytné ve správě toku dat nastavit jako výstup vytvořený datový sklad a zadat název schématu spolu s přístupovými údaji vytvořenými pro ETL službu.

3.2.1 Implementace záložního úložiště

Pro potřeby zálohy je vytvořen bucket v úložišti Amazon S3, kde ETL systém zasílá poslaná data z aplikace.

Založení S3 úložiště je podstatně jednodušší na vytvoření, jelikož je potřeba vyplnit pouze název bucketu, zvolit oblast kde bude bucket uložen a vygenerovat přístupový klíč (*AWS Access Key ID*) a tajný přístupový klíč (*AWS Secret Access Key*) ve správě uživatelů.

Tyto klíče jsou následně využity při nastavování zálohy ve službě Alooma. Spolu s vygenerovanými klíči se zvolí také prefix, jež zpřehlední ukládání dat v S3. Po napojení S3 na ETL proces, má záloha dat z iOS aplikace následující adresu *s3://alooma_spolecnost/alooma_backup/iOS_app/YYYY-mm-DD-HH-MM-SS*.

Data z aplikace Android mají pak analogickou adresu, pouze se změní adresář, kde je soubor uložen. Výsledný název souboru se zálohou má mít následující tvar *s3://alooma_spolecnost/alooma_backup/Android_app/YYYY-mm-DD-HH-MM-SS*.

3.3 Napojení mobilní aplikace

Postup u napojení mobilní aplikace je odlišná pro obě mobilní platformy (Android a iOS). Přičemž každá z platforem má také vlastní SDK. Obě SDK vycházejí z knihovny Mixpanelu.

Při napojení aplikace je potřeba vytvořit vstupní zdroj ve službě Alooma pro každou aplikaci zvlášť. Při vytváření vstupu je žádanou podmínkou pouze jméno, pod jakým bude v ETL systému veden a poté zvolit pro jaký operační systém bude zdroj použit. Po vytvoření zdroje se vygeneruje token (unikátní klíč), který je využit při zasílání událostí v aplikaci.

3.3.1 Android

Pro správnou funkcionalitu je v manifestu nezbytné aplikovat oprávnění INTERNET, ACCESS_NETWORK_STATE a BLUETOOTH. Při každé inicializaci SDK je zvolána metoda, kde jsou jako parametry uvedeny vygenerovaný token a host inputs.alooma.com.

Pro odeslání události se nejdříve vytvoří soubor JSON s parametry, který se následně připraví k odeslání pomocí metody track().

```
JSONObject attr = new JSONObject();
attr.put("os","Android");
attr.put("manufacturer", "Samsung");
attr.put("model", "G960F");
attr.put("os_version", "23");
attr.put("wifi", "true");
attr.put("screen_dpi", "400");
attr.put("screen_width", "1440");
attr.put("screen_height", "2960");
```

```
attr.put(“price”:”1234”
mAPI.track("SearchItemSelected", attr);
```

3.3.2 iOS

Na je iOS postup obdobný, kdy se inicializuje knihovna spolu s nastavením tokenu a adresy hosta. Následně se při každé komunikaci zavolá metoda *Alooma *alooma = [Alooma sharedInstance];* která nachystané soubory se zachycenými událostmi odešle.

```
[alooma track:@" SearchItemSelected" properties:@{
@"os": @"iOS",
@"manufacturer": @"Apple",
@"model": @"iPhone 8",
@"os_version": @"11.2.3",
@"wifi": @"true",
@"screen_width": @"320",
@"screen_height": @"568"
@”price”: @”1234”
}];
```

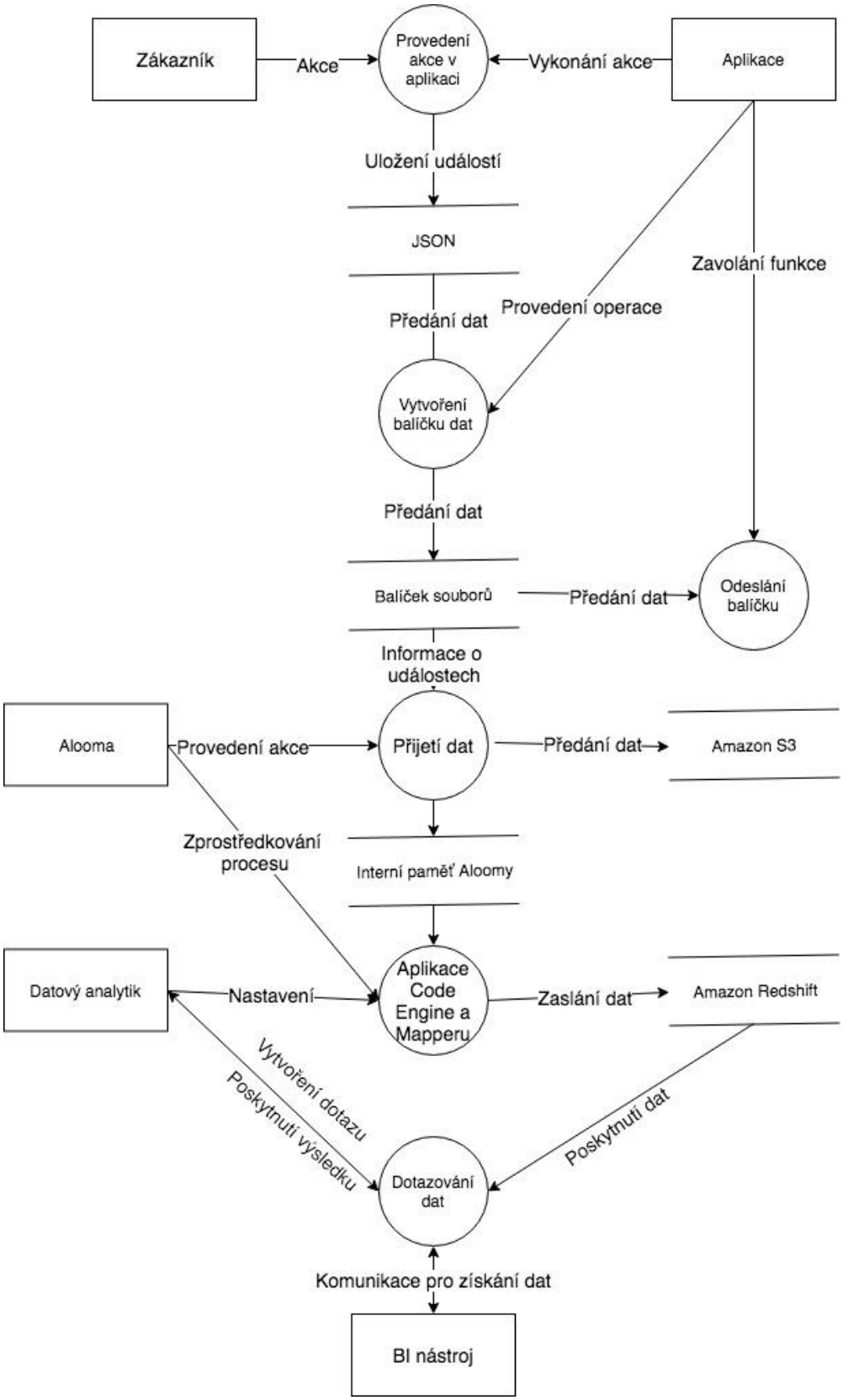
3.3.3 Hashování

Z důvodu připravované směrnice GDPR obsahuje samotná aplikace funkci pro převod řetězce do podoby hashe. Jako parametr funkce slouží například email, který je jednoznačným identifikátorem uživatele. Tento prvek není možné implementovat až ve fázi ETL a to z důvodu záložního odesílání neupravených záznamů do Amazon S3. Nastal by tedy stav, kdy by v cílovém datovém skladu byl formát dat v korektní formě (dle směrnice GDPR), ovšem záloha by obsahovala explicitně uvedený email. Pro obě platformy je dostupná knihovna s hashovací funkcí *SHA256*. Využitím shodné hashovací funkce na obou platformách je zajištěn stejný výsledek a zároveň je možné data párovat na základě daného atributu, aniž by byla odhalena jeho totožnost.

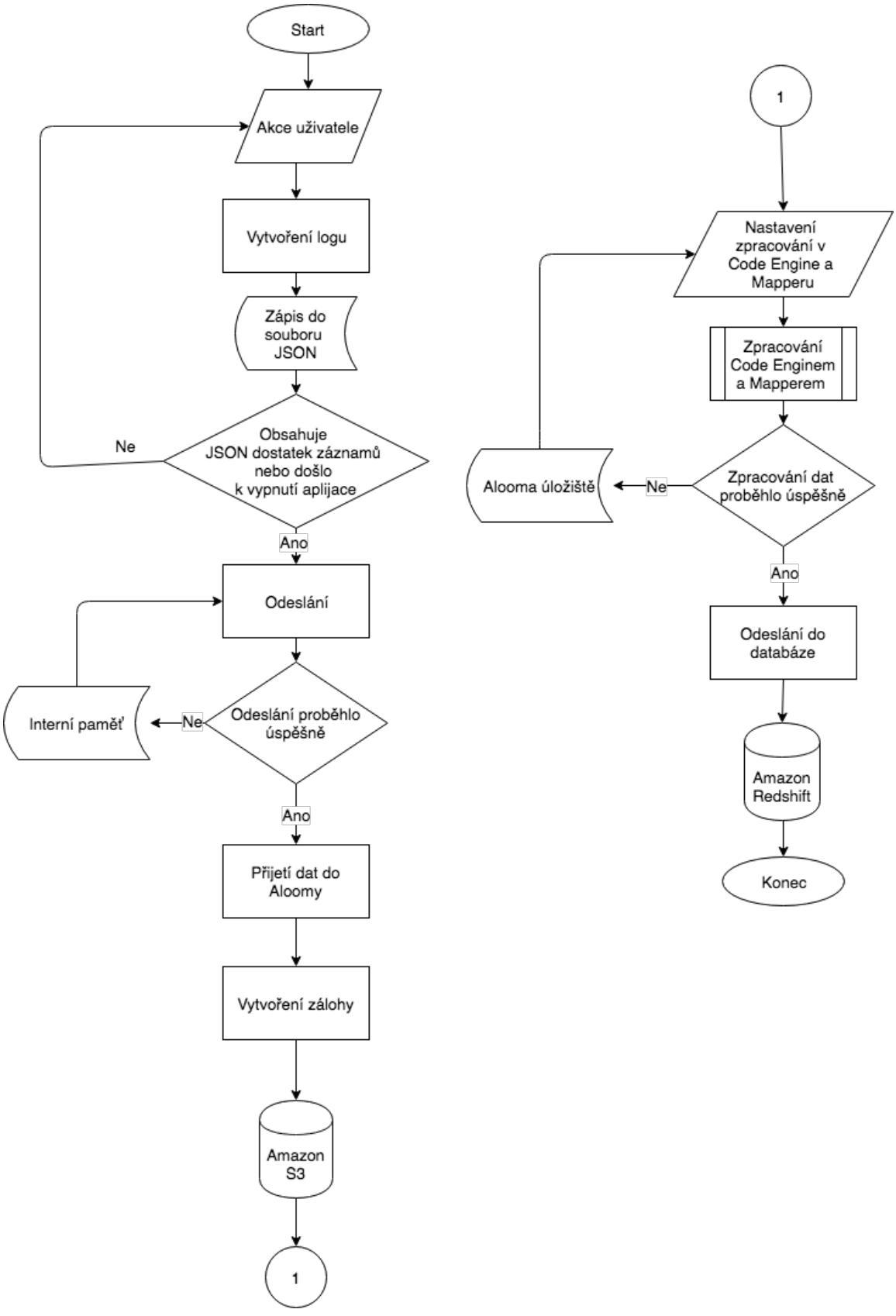
3.4 Sběr událostí

Prvním podnětem pro zaznamenání události je vyvolání události uživatelem aplikace, resp. vykonání kódu. V takovém případě se vytvoří soubor JSON. Aplikace neposílá každý soubor ihned, ale jednotlivé soubory sbírá a po shromáždění dostatečného počtu je hromadně odešle. Pokud je zařízení offline, pak se balíčky uloží do interní paměti zařízení a pokouší se operaci opakovat.

Pokud vše proběhne bez problémů je balíček odeslán na API rozhraní služby Alooma. Ihned po přijetí se data automaticky odešlou ve formátu GZip do záložního úložiště Amazon S3. Následně se zpracují dle požadovaného nastavení nástroji Code *Engine* a *Mapper*. V případně, kdy dojde k chybě v jednom z procesů, uloží se záznamy do interního úložiště Aloomy a po provedení změn podle příslušného chybového hlášení je možné akci opakovat. Projde-li však událost celým ETL procesem bez problému, je odeslána do cílového úložiště Redshift, kde je uložena do příslušné tabulky a provedeno vytvoření nebo aktualizace záznamů v tabulkách dimenzí.



Obrázek č. 9: DFD diagram toku dat při sběru událostí z aplikace.
(Zdroj: Vlastní zpracování)



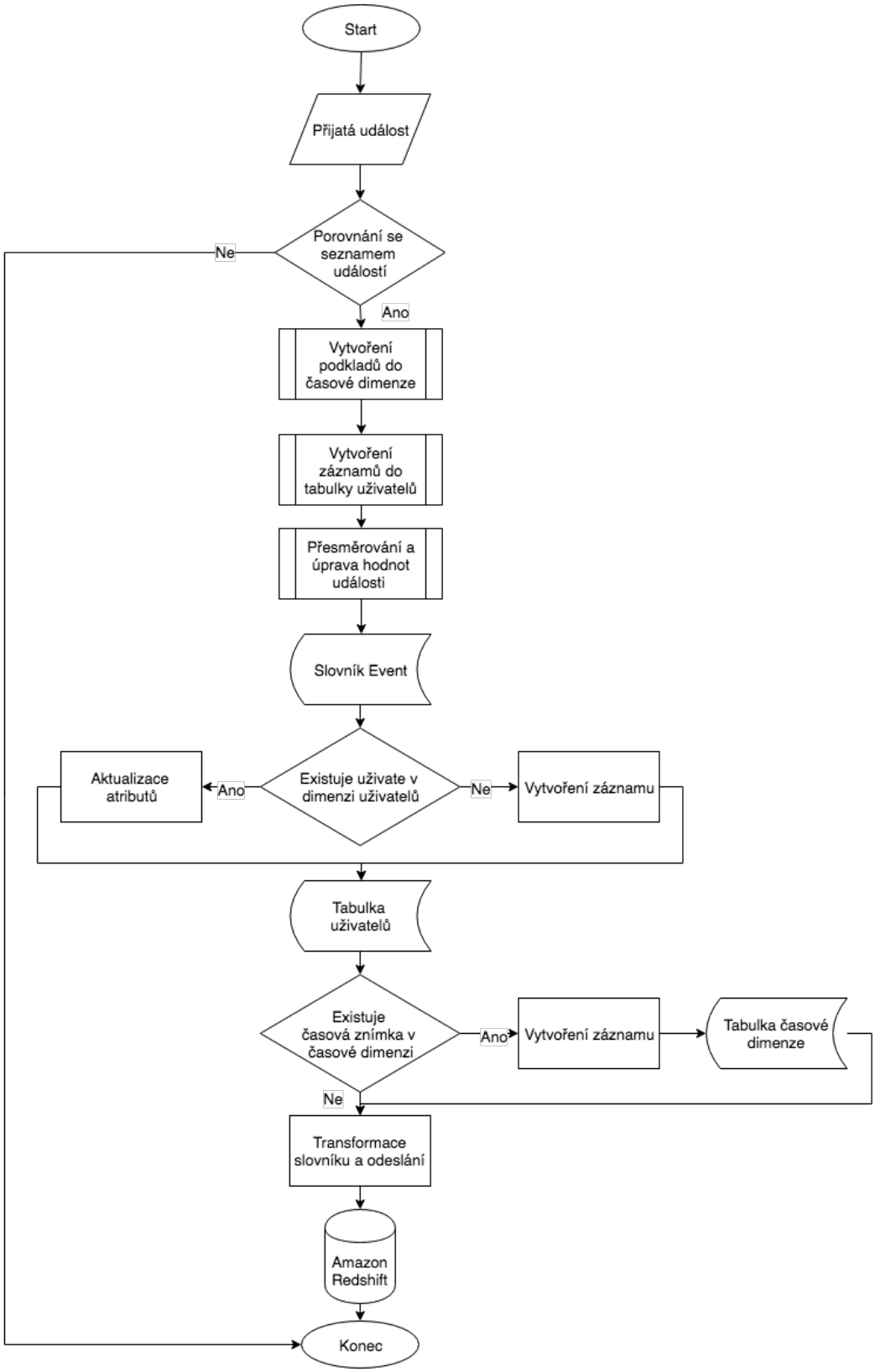
Obrázek č. 10: Vývojový diagram sběru událostí z aplikace.
(Zdroj: Vlastní zpracování)

3.5 Transformace událostí

Přijatá událost je ve formě slovníku[6] předána nástroji *Code Engine*. Zde je nadefinován další slovník, jenž má jako klíče název události malými písmeny a jako hodnoty daných klíčů jsou pak definované požadované názvy. Na začátku funkce jsou porovnány názvy událostí právě s tímto slovníkem. Pokud je událost obsažena, je předána k dalšímu zpracování, v opačném případě se s touto událostí nepracuje a je zahozena (událost je však obsažena v záloze). Následně je na základě časové známky vytvořen slovník s podklady pro časovou dimenzi (den v týdnu, číslo týdne apod.) Obdobně je sestaven slovník s údaji o uživateli. Na závěr je událost přejmenována a podle jména je také určena tabulka, kde se událost uloží.

Nástroj *Mapper* je nastaven na datové typy, jež jsou shodné s tabulkou vytvořenou v datovém skladu. Avšak pro správnou funkčnost datového skladu bylo nutné nastavit mapování událostí pro časovou a uživatelskou dimenzi, kde jsou data zároveň ukládány do dvou tabulek. První je *_log* tabulka, kde jsou uloženy všechny záznamy a poté tabulka *Date_dimension* a *User_dimension*, kde jsou jen unikátní záznamy dané primárním klíčem. *Mapper* tedy při příchodu zaregistrování záznamu zkontroluje a zda je záznam z tabulky *_log* obsažen v tabulce dimenzí, pokud ano, pak v případě uživatele aktualizuje atributy. U časové tabulky se pouze vkládají nové záznamy, není důvod k aktualizaci záznamů. Na závěr jsou data odeslána do datového skladu.

[6] Datová struktura používaná v jazyce Python [25].



Obrázek č. 11: Vývojový diagram ETL procesu – transformace události.
(Zdroj: Vlastní zpracování)

3.6 Schéma datového skladu

Datový sklad má schématickou podobu souhvězdí. Každá událost bude tvořena jednou tabulkou faktů, přičemž dimenze bude tvořit časová tabulka a tabulka uživatelů.

Alooma ve výchozím nastavení vytvoří jednu tabulku v cílovém datovém skladu na jeden datový vstup, což je neefektivní a postrádá to smysl funkce datového skladu. Prvním problémem je obrovský počet řádků, což by se v případě ponechání toho nastavení negativně promítlo na výkonu. Druhou nevýhodou by bylo velké množství zbytečných sloupců (obsahovaly by hodnoty NULL), jelikož všechny události mají shodnou jen základní sadu atributů a poté jsou atributy voleny dle kontextu (u nákupu například cena, naopak u přihlášení to může být typ). Posledním problémem je dělení dat podle zdrojů, což by znamenalo rozdílné tabulky pro Android i iOS, případně web.

Z toho důvodu je nutné vytvořit funkci v nástroji *Code Engine*, jenž je součástí Aloomy. Funkce má za úkol poslanou událost přesměrovat na požadovanou tabulku (shodnou s názvem, události), přičemž cílové umístění se definuje hodnotou přiřazenou k atributu s názvem *event_type*.

3.6.1 Časová dimenze

Časová dimenze bude sloužit jako popis času pro jednotlivé tabulky faktů. Bude generována na základě zaslaného času vytvoření události. API pro zasílání událostí do ETL systému generuje dvě časové známky. První je tzv. *sending_time*, která určuje, kdy byla událost odeslána, což je poněkud nepřesné z důvodu čekání dat ve frontě na odeslání. Tento efekt se ještě umocňuje, je-li zařízení offline. Proto je zvolen *time*, který odpovídá reálnému vytvoření události.

Na základě této časové známky se v *Code Enginu* generují tyto hodnoty:

- Kvartál – číslo kvartálu v rámci roku. Nabývá hodnot v rozsahu 1–4.
- Den v týdnu (název) – anglický název pro den v týdnu.
- Den v týdnu (číselná hodnota) – stejné jako předchozí, avšak název je nahrazen číslicí 1-7.
- Číslo týdne – číslo týdne v rámci roku. Má dvě varianty, jedna zahrnuje pouze číslo týden a druhá je ve spojení s rokem.

- Datum – datum v anglickém formátu – YYYY-MM-DD.
- Hodina – označení hodiny – nabývá hodnot 00-23.
- Minuta – obdobný případ jako hodina ovšem s rozsahem 00-59.
- Časová známka – časová známka.

3.6.2 Uživatelská dimenze

Bude sloužit pro popis jednotlivých uživatelů. Tabulka uživatelů v sobě bude agregovat základní informace o uživateli, jež se budou aktualizovat při každém zaznamenání nové události.

Základem bude ID, jež se generuje automaticky ETL systémem v návaznosti na instalaci.

Uživatelská dimenze bude obsahovat následující hodnoty:

- ID instalace – generuje se pomocí API služby Alooma.
- Časová známka první zaznamenané události – nelze párovat data z obchodů pro distribuci aplikací a jednotlivými daty z instalace. Proto bude sloužit jako náhrada času instalace.
- Časová známka poslední zaznamenané události – pro filtrování poslední aktivity uživatele.
- Email (hash) – emailová adresa (pokud se u uživatele zaznamenala) v podobě hashe. Možnost získat z událostí *Login Success* nebo *Purchase Completed*.
- Jazyk – poslední známá konfigurace jazyka zařízení ve formátu ISO 639-1.
- Region – poslední známá konfigurace regionu zařízení ve formátu ISO 3166 (Alpha-2).

3.6.3 Tabulky faktů

Jak již bylo zmíněno, jedna tabulka faktů odpovídá jedné události. Každá událost je specifická a obsahuje kontextové informace. Základem každé události je sada následujících informací:

- ID události – hash generovaný službou Alooma.
- Jazyk – konfigurace jazyka zařízení ve formátu ISO 639-1.

- Region – konfigurace regionu zařízení ve formátu ISO 3166 (Alpha-2).
- Měna – aktuální nastavení měny v aplikaci ve formátu ISO 4217 (Code Alpha).
- Zařízení – kódové označení zařízení.
- Výrobce – název výrobce zařízení.
- Model – model zařízení.
- Operační systém – operační systém zařízení.
- Verze operačního systému – verze operačního systému (u Androidu API úroveň, iOS ve formátu x.x.x.x)
- Typ – typ zařízení (tablet/telefon).
- Rozlišení – rozlišení displeje zařízení (výška, šířka).
- Verze aplikace – verze aplikace ve formátu x.x.x.
- ID instalace – generuje se pomocí API služby Alooma, používáno jako ID uživatele.
- Dostupnost Wifi – informace o využití Wi-Fi technologie při zaznamenání události.
- _metadata – pro potřeby služby Alooma a identifikace tabulky.
- Input – název vstupu, na němž byla událost zaznamenána.
- Klient – název klienta.
- Čas vytvoření – časová známka s údajem kdy byla zaznamenána událost.
- Čas zaslání – časová známka zaslání události.
- Název události – název události ze slovníku.

Kontextové informace každé tabulky odpovídají jejich informační funkci a relevanci dat v rámci umístění události. Pro potřeby této práce je vytvořeno pět tabulek.

První tabulkou je *AppLaunched*, která se zaznamená pří spuštění aplikace. Počet výskytů události v kombinaci s použitím SQL příkazu *DISTINCT(Count(ID instalace))*, udává počet jedinečných uživatelů, kteří aplikaci spustili. Tabulka s událostí obsahuje pouze sadu základních informací.

SearchItemSelected obsahuje informace o výběru produktu z vyhledávače, dle zadaného filtru. Událost se vyvolá při kliku na konkrétní položku z nabídky. Kromě základních informací obsahuje také:

- Originální cenu – hodnota produktu v eurech.

- Formátovaná cena – jedná se o cenu, která je zobrazena uživateli v závislosti na zvolené měně v aplikaci. Pokud je zvolená měna euro, pak je formátovaná cena shodná s originální.
- Nastavení filtru – jedná se o množinu podobných atributů, jenž označují jednotlivé položky filtru a jejich hodnoty.

Při potvrzení vybraného produktu ze seznamu se uživatel dostane na stránku pro rezervaci produktu. Události se zapisují do tabulky *PurchaseInitialzed*. Opět obsahují všechny základní atributy a prakticky shodné atributy s tabulkou *SearchItemSelected*, takže jde o:

- Originální cenu – hodnota produktu v eurech.
- Formátovaná cena – jedná se o cenu, která je zobrazena uživateli v závislosti na zvolené měně v aplikaci. Pokud je zvolená měna euro, pak je formátovaná cena shodná s originální.

Po dokončení rezervace se vyvolá událost, která se zapíše do tabulky *PurcaseCompleted*. Kromě základních atributů obsahuje také:

- SandBox – pro určení, zda jde o testovací rezervaci nebo reálnou. Při vytváření analýz je tedy nutné vzít v potaz jen ty záznamy, kde je atribut sandBox nastaven na false (jedná se o boolean).
- Použití promo kódu – jedná se o boolean, který označuje, zda byl v rámci rezervace zadán promo kód.
- Počet položek – označuje počet zakoupených položek v rámci jedné rezervace.
- ID nákupu – ID nákupu je identifikátor nákupu vrácený serverem.
- Originální cenu – hodnota produktu v eurech.
- Formátovaná cena – jedná se o cenu, která je zobrazena uživateli v závislosti na zvolené měně v aplikaci. Pokud je zvolená měna euro, pak je formátovaná cena shodná s originální.
- Cena služeb – pokud jsou zakoupeny také doplňkové služby je cena těchto služeb reflektována v tomto atributu. Cena je uváděna v eurech.
- Celková cena – celková cena je celková suma v eurech, jež zákazník zaplatí.

- Platební metoda – výchozí platební metodou je kreditní karta, ovšem je možné zvolit i alternativní možnosti. Zvolená metoda při provedení nákupu je zaznamenána v tomto atributu.
- Email – kontaktní email zadaný při rezervaci, resp. jeho hash podoba.
- Národnost – jelikož si produkty v aplikace může koupit kdokoli, je zaznamenána také národnost kupujícího.

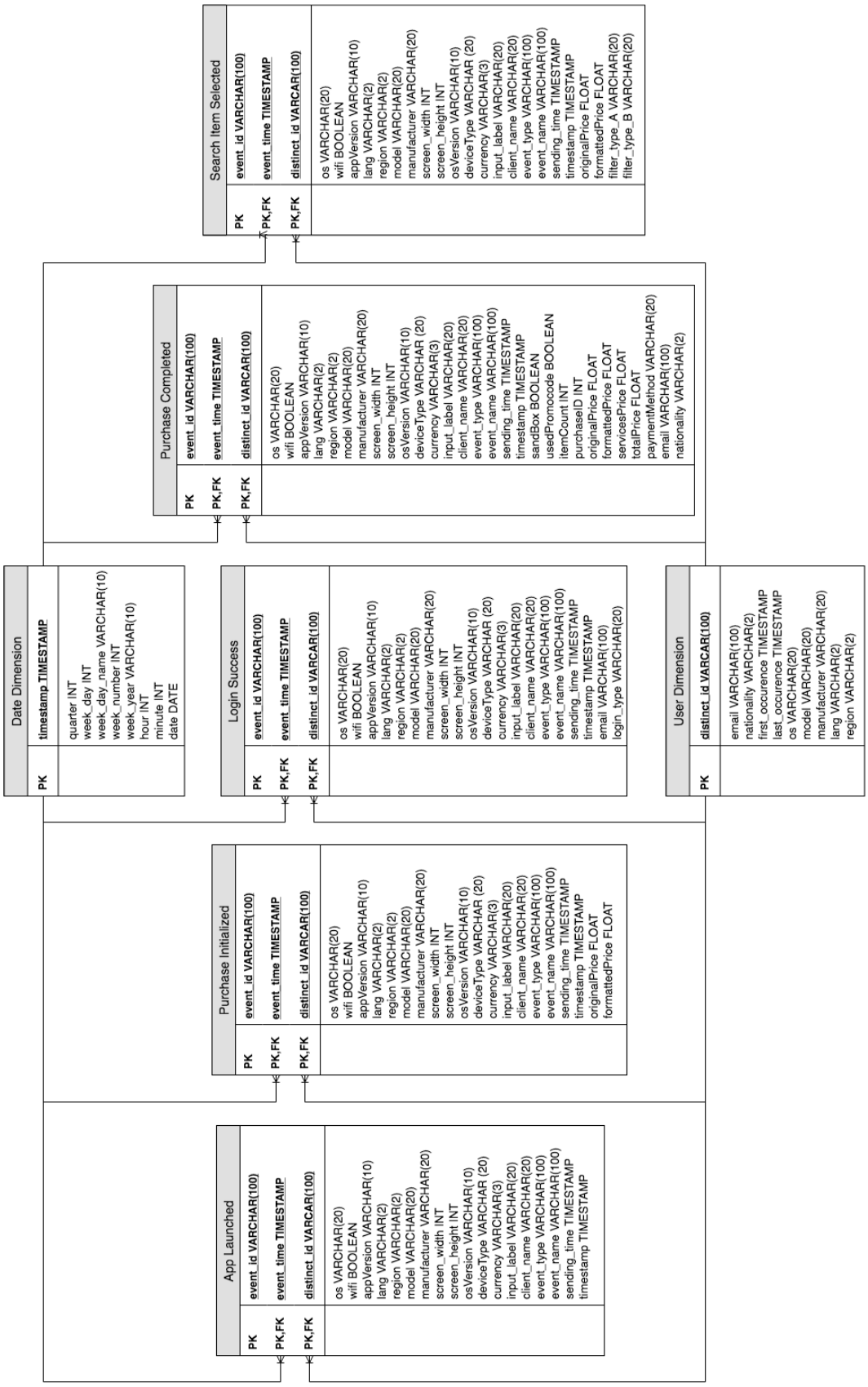
Na závěr je zde událost vyvolaná při úspěšném přihlášení – *LoginSuccess*, která má mimo společné atributy také informace o:

- Emailu – kontaktní email zadaný při rezervaci, resp. jeho hash.
- Typ – jedná se o typ přihlášení, který uživatel zvolil. Může se jednat o výchozí možnost emailu nebo také pomocí některé z podporovaných sociálních sítí.

3.6.4 Určení primárních a cizích klíčů

V dimenzionálních tabulkách se vyskytuje jednoduchý primární klíč, který je tvořen jen jedním atributem. V případě časové dimenze je tímto atributem časová známka. U uživatelské dimenze je pak primární klíč tvořen identifikátorem instalace, který je tvořen automaticky ETL službou.

U tabulek faktů je primární klíč složený, obsahuje tedy více atributů. Konkrétně se jedná o ID události a ID instalace (obě generovány ETL službou) a časová známka zaznamenání události. ID instalace a časová známka události byly zmíněny v předchozím odstavci jako primární klíče jednotlivých tabulek dimenzí, což znamená, že kromě funkce primárního klíče plní také funkci cizího klíče v tabulkách faktů a zajišťují tak relační vazbu mezi tabulkami dimenzí a faktů.



Obrázek č. 12: Entito-relační diagram datového skladu.
(Zdroj: Vlastní zpracování)Vytvoření datového skladu

Pomocí ETL bohužel není možné vytvořit relace mezi jednotlivými tabulkami, ani určit primární a cizí klíče v tabulce. Pro každou nově nadefinovanou událost je tedy nutné vytvářet tabulky pomocí dotazovacího jazyka SQL, kde se nadefinují primární a cizí klíče.

Příkaz pro vytvoření tabulky *SearchItemSelected* bude vypadat následovně:

```
CREATE TABLE SearchItemSelected(
event_id VARCHAR(100) NOT NUL,
event_time TIMESTAMP NOT NULL,
distinct_id VARCAR(100) NOT NULL,
os VARCHAR(20),
wifi BOOLEAN,
appVersion VARCHAR(10),
lang VARCHAR(2),
region VARCHAR(2),
model VARCHAR(20),
manufacturer VARCHAR(20),
screen_width INT,
screen_height INT,
osVersion VARCHAR(10),
deviceType VARCHAR (20),
currency VARCHAR(3),
input_label VARCHAR(20),
client_name VARCHAR(20),
event_type VARCHAR(100),
event_name VARCHAR(100),
sending_time TIMESTAMP,
timestamp TIMESTAMP,
originalPrice FLOAT,
formattedPrice FLOAT,
PRIMARY KEY(event._id, event._time, distinct_id),
FOREIGN KEY(event_time) REFERENCES  Date_Dimension(timestamp),
FOREIGN KEY(event_time) REFERENCES User_Dimension(distinct_id)
```

)

DISTKEY(event_time)

COMPOUND SORTKEY(event_time,os)

Příkazy DISTKEY je aplikován pouze na jeden sloupec a udává, jak budou hodnoty tříděny mezi jednotlivými uzly (zaručí, že řádky se stejnými hodnotami zůstanou na stejném uzlu). SORTKEY pak upřesňuje, jak budou data na jednotlivých uzlech řazena. Celý princip je podobný indexaci, což je běžně dostupná funkce v SQL.

Z příkladu je patrné, že DISTKEY byl aplikován na časovou známku, protože s časovým údajem se pracuje při téměř každém dotazu a na rozdíl od uživatelského ID (hash) nebo event_id (hash) je také tříditelné. Podobná situace je také u SORTKEY, kde navíc ovšem přibylo také řazení na základě operačního systému, což je také hojně využívaný prvek u prováděných analýz.

3.6.5 Správa uživatelů

Do datového skladu budou mít přístup všichni analytici společnosti, ale také další pracovníci. Z důvodu bezpečnosti a integrity dat je možné vystavit jen účty pro čtení tzv. Read-only. Vytvoření uživatele je docíleno příkazem sadou příkazu pro vytvoření uživatele a udělením oprávnění *USAGE* na schéma *alooma*.

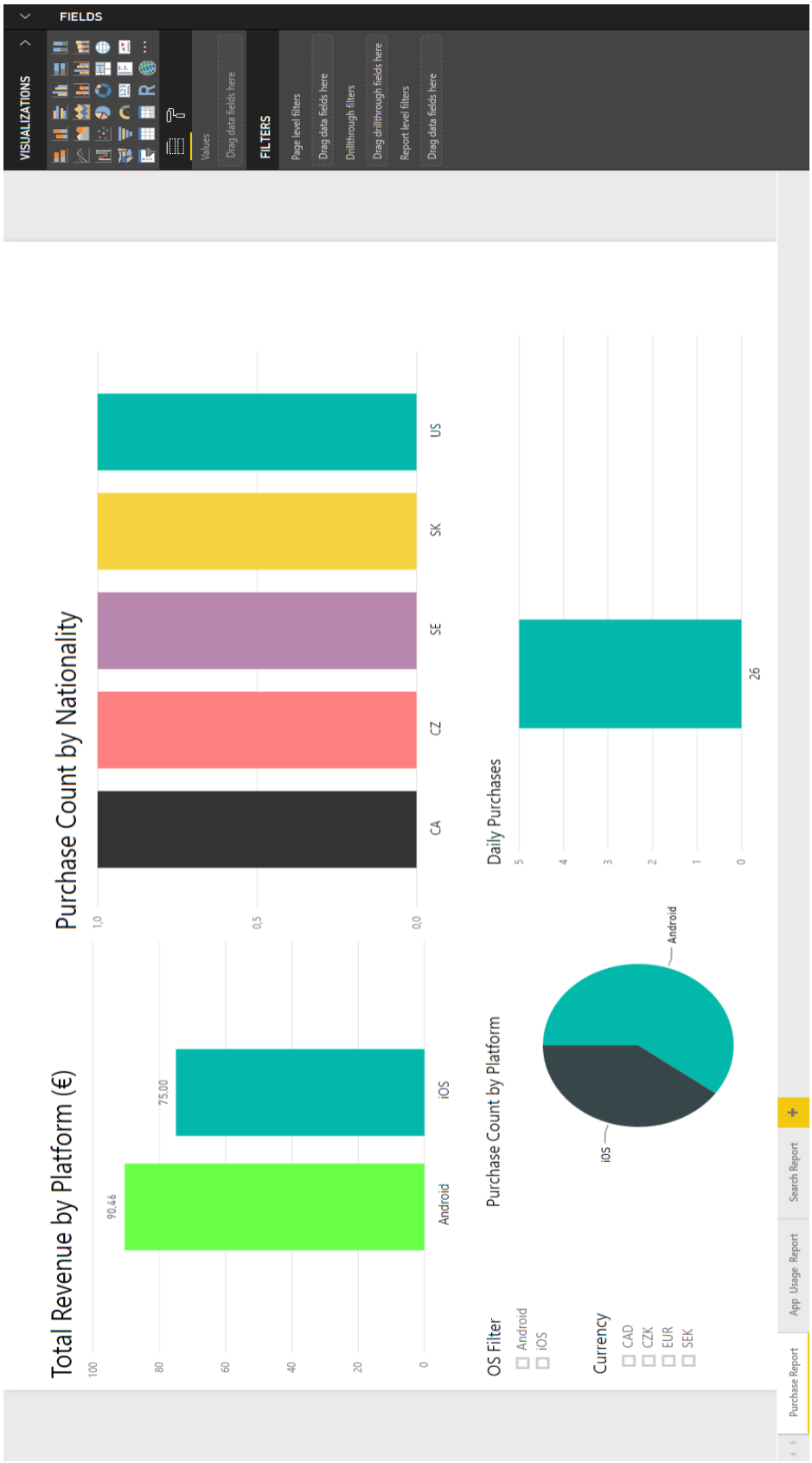
3.7 Prezentace dat

Uložená data se budou využívat k vytváření reportů pro potřeby reportování vedení, vyhodnocení zavedených nových funkcí nebo identifikaci možných problémů. Pro technicky zdatné uživatele není problém, po přidělení přístupu, dotazovat se přes jakýkoli SQL klient, který podporuje ODBC nebo JDBC ovladače. Ovšem taková data jsou velice strohá.

Pro prezentaci je ideální použít některý z Business Intelligence nástrojů, jenž umožňují přehledně prezentovat požadovaná data, a manipulaci s nimi zvládnou také méně zdatní uživatelé, kteří data mohou snadno filtrovat pomocí přednastavených filtrů.

Jedním z možných řešení je Microsoft Power BI, který podporuje Amazon Redshift a umožňuje také snadnou obsluhu.

Příklad reportu (Obrázek č. 13: Příklad reportu v Power BI založeného na tabulce PurcaseCompleted.) je založen na tabulce *PurcaseCompleted*. V reportu je znázorněn graf s celkovými příjmy z prodaných, jenž vychází ze sumy atributu *OriginalPrice*. Následuje sloupcový graf se znázorněním počtu nákupů dělených podle národnosti zadané při rezervaci produktu, která je dána atributem *Nationality*. Koláčový graf znázorňuje poměr nákupů na jednotlivých platformách podle atributu *OS*. Na závěr je zde opět sloupcový graf s počtem nákupů v jednotlivé dny, který vychází z časové známky. V reportu jsou dostupné také základní filtry pro volbu operačního systému a měny, po jejichž aktivaci se změna promítne ve všech grafech v rámci daného reportu.



Obrázek č. 13: Příklad reportu v Power BI založeného na tabulce PurcaseCompleted.
(Zdroj: Vlastní zpracování)

3.8 Zhodnocení

Společnost za projekt vynaloží jednorázově 34 500 Kč, což jsou náklady pro pokrytí implementační fáze projektu. Do těchto nákladů patří plat programátora a datových analytiků, kteří se na projektu podíleli. Na závěr je nutné zahrnout také náklady na export dat ze starého reportovacího systému.

Další náklady představují výdaje služeb zajišťující ETL proces a provoz datového skladu. Tyto náklady jsou placeny cyklicky (v tomto případě jsou uváděny na měsíční bázi), až na službu Alooma, která má financování formou kreditů, kdy jeden kredit představuje určitý počet přenesených událostí a není nijak časově omezen. Nelze tedy přesně určit velikost výdajů spojených s touto službou, protože vždy bude odvozena od aktuální měsíční spotřeby.

Náklady na implementaci jsou odvozeny od hodinové sazby pracovníků. Operace a jejich finanční náročnost je uvedena v následující tabulce.

Tabulka č. 20: Přehled implementačních nákladů.
(Zdroj: Vlastní zpracování)

Popis činnosti	Počet hodin	Sazba	Celkem
Analýza dostupných produktů	20	300	6 000 Kč
Implementace do aplikace	5	500	2 500 Kč
Návrh a vytvoření datového skladu	10	300	3 000 Kč
Navržení a implementace ETL procesu	20	300	6 000 Kč
Testování	10	200	2000 Kč
Export dat	30	500	15 000 Kč
Celkem			**34 500 Kč**

Další část nákladu tvoří zmíněné náklady na provoz služeb spolu s náklady na datového analytika, který se bude starat o údržbu systému a připravovat podklady pro nové události. Cena databáze je uvažována při využití nejvýhodnějšího tarifu spolu s volbou dvou výpočetních uzlů, a to z důvodu většího výkonu.

Tabulka č. 21: Přehled nákladů na provoz za měsíc (kurz dle ČNB.cz ze dne 21. 4. 2018).
(Zdroj: Vlastní zpracování)

Služba	Cena USD	Cena CZK (≈)
Amazon Redshift	$280	5 910 Kč
Amazon S3	$34	700 Kč
Alooma	$500	10 550 Kč
Údržba datového skladu a ETL	-	6 000 Kč
Celkem		**23 160 Kč**

Náklady na ETL službu nebudou konstantní, ale platí se podle toho, kolik událostí je v rámci daného období zaznamenáno. Predikce vychází z plynulých dvanácti měsíců a v závislosti na nich byla vytvořena regresní funkce, která nejlépe vystihovala trend.

Graf č. 2: Předpokládaný vývoj ceny ETL služby.
(Zdroj: Vlastní zpracování)

3.8.1 Přínosy práce

Přínosy pro navržené řešení se nedají jednoznačně kvantifikovat a jde tedy o přínosy neměřitelné. Mezi tyto přínosy patří především:

- Správa vlastních dat – z pohledu společnosti je obrovskou výhodou správa vlastních dat. To společnosti umožňuje s daty flexibilně nakládat a používat je téměř kdykoli a odkudkoli. Redshift podporuje přístup pomocí standardizovaných ovladačů ODBC a JDBC, dotazy tedy mohou být prováděny v rámci velkého množství aplikací, ale také využity v nejrůznějších firemních reportovacích systémech. Datový sklad u Amazonu také společnosti poskytuje určitou flexibilitu a umožňuje jí experimentovat s volbou služeb pro ETL procesy nebo BI nástroji, pokud by se některý z nich neosvědčil, bez rizika ztráty dat.
- Zavedení ETL procesu – v současném řešení je znatelná absence ETL procesu. Data pak postrádají konzistenci, čímž je většina času při vytváření reportů věnována zkoumání, úpravě a přepočtu získaných hodnot. Zavedení ETL procesu bude mít tedy pozitivní vliv na ušetřený čas.

Závěr

Cílem této diplomové práce byl návrh datového skladu pro potřeby ukládání událostí vytvořených v mobilní aplikaci, který nahradí současně používané řešení. Cíl práce se mi povedlo splnit implementací datového skladu spolu s ETL službou, která byla otestována a je připravena k hromadnému nasazení.

V první části práce byly popsány a vysvětleny pojmy, s kterými se čtenář při čtení této práce setká a jenž jsou důležité pro celkové pochopení kontextu práce. Následně byla provedena analýza, kde se zhodnotil současné používaný systém a pomocí SWOT analýzy se vyhodnotily jeho silné a slabé stránky, hrozby, ale také příležitosti, které by mohlo zavedení nového úložiště přinést. Součástí této kapitoly byl také popis vybraných produktů, které jsou v současné době dostupné na trhu, spolu s identifikací jejich vlastností jako jsou podporované datové typy nebo jejich omezení.

V samotném návrhu řešení byly v závislosti na výsledcích analytické kapitoly zvoleny řešení, která budou implementována. Implementace začala nastavením datového úložiště a vytvořením datového toku v ETL službě. Následně se provedlo napojení mobilních aplikací. Pro datový sklad bylo vytvořeno schéma, které je znázorněno na relačním diagramu, a v ETL službě vytvořen skript pro úpravu příchozích dat. K popisu komunikace a transformace dat bylo využito metod datového modelování, a to konkrétně DFD diagramu a vývojového diagramu. Nově implementované řešení bylo odzkoušeno a byl vytvořen report na základě nasbíraných dat. Na závěr kapitoly byly vyhodnoceny náklady celého projektu, odhad cen do následujících měsíců a také identifikovány přínosy, jež byly dosaženy zavedením datového skladu.

Seznam použitých zdrojů

[1] LACKO, Ľuboslav. *Business Intelligence v SQL Serveru 2008: reportovací, analytické a další datové služby*. Brno: Computer Press, 2009. ISBN 978-80-251-2887-9.

[2] KIMBALL, Ralph a Margy ROSS. *The data warehouse toolkit: the definitive guide to dimensional modeling*. Third edition. Indianapolis, IN: John Wiley & Sons, 2013. ISBN 978-1-118-53080-1.

[3] PONNIAH, Paulraj. *Data warehousing fundamentals for IT professionals*. 2nd ed. Hoboken, N.J.: John Wiley, 2010. ISBN 978-0-470-46207-2.

[4] The Star Schema Data Warehouse Design. *Pitney Bowes* [online]. Stamford: Pitney Bowes, ©2018 [cit. 2018-05-11]. Dostupné z: http://support.pb.com/help/spectrum/9.0/webhelp/en/EnterpriseDataIntegration/EnterpriseDataIntegration /source/Introduction/StarSchemaConcept.html

[5] The Main Weakness of Snowflake Schemas. *Data Warehousing, BI and Data Science* [online]. Londýn: Data Warehousing, BI and Data Science, ©2018 [cit. 2018-05-11]. Dostupné z: https://dwbi1.wordpress.com/2012/07/16/the-main-weakness-of-snowflake-schemas/

[6] BEAULIEU, Alan. *Learning SQL*. 2nd ed. Sebastopol: O'Reilly, c2009. ISBN 978-0-596-52083-0.

[7] KOCH, Miloš a Bernard NEUWIRTH. *Datové a funkční modelování*. Vyd. 4., rozš. Brno: Akademické nakladatelství CERM, 2010. ISBN 978-80-214-4125-5.

[8] WODEHOUSE, Carey. *SQL vs. NoSQL: What's the Difference?*. Upwork Global Inc [online]. Upwork Global Inc, ©2018 [cit. 2018-05-11]. Dostupné z: https://www.upwork.com/hiring/data/sql-vs-nosql-databases-whats-the-difference/

[9] SHAFRANOVICH, Yakov. *Common Format and MIME Type for Comma-Separated Values (CSV) Files*. IETF [online]. IETF, ©2005 [cit. 2018-05-11]. Dostupné z: http://www.ietf.org/rfc/rfc4180.txt

[10] CSV – Comma Separated Values. *Data Hub* [online]. Data Hub, ©2018 [cit. 2018-05-11]. Dostupné z: https://datahub.io/docs/data-packages/csv

[11] Data Warehousing. *Data Pals* [online]. Data Pals, ©2015 [cit. 2018-05-11]. Dostupné z: http://datapals.com/article_post/data-warehousing/

[12] ECMA, International. *The JSON Data Interchange Syntax* [online]. Ecma International, 2017 [cit. 2018-05-11]. Dostupné z: http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf

[13] GRAHAM-SMITH, Darien. Weekend Project: Build your own supercomputer. In: *PCAuthority.com.au* [online]. 29. 6. 2012 [cit. 2018-05-11]. Dostupné z: https://www.pcauthority.com.au/feature/weekend-project-build-your-own-supercomputer-306972

[14] Amazon Simple Storage Service Documentation. *Amazon Web Services* [online]. Amazon Web Services, ©2018 [cit. 2018-05-11]. Dostupné z: https://aws.amazon.com/documentation/s3/

[15] ROUSE, Margaret. MPP database (massively parallel processing database). In: *SearchDataManagement.TechTarget.com* [online]. 1. 6. 2015 [cit. 2018-05-11]. Dostupné z: https://searchdatamanagement.techtarget.com/definition/MPP-database-massively-parallel-processing-database

[16] Azure SQL Data Warehouse: Massively parallel processing (MPP) architecture. *Microsoft Azure* [online]. Microsoft, 17. 4. 2018 [cit. 2018-05-11]. Dostupné z: https://docs.microsoft.com/en-us/azure/sql-data-warehouse/massively-parallel-processing-mpp-architecture

[17] FLYDATA TEAM. Introduction to Massively Parallel Processing. In: *FlyData.com/blog* [online]. 29. 4. 2015 [cit. 2018-05-11]. Dostupné z: https://www.flydata.com/blog/introduction-to-massively-parallel-processing/

[18] Open Handset Alliance. Overview. *Open Headset Alliance* [online]. Open Headset Alliance, ©2018 [cit. 2018-05-11]. Dostupné z: http://www.openhandsetalliance.com/oha_overview.html

[19] Android Developers. *Google Developers* [online]. Google Developers, ©2018 [cit. 2018-05-11]. Dostupné z: http://developer.android.com/sdk/index.html

[20] GARGENTA, Marko a Masumi NAKAMURA. *Learning Android*. Second edition. Sebastopol, CA: O'Reilly Media, 2014. ISBN 978-1-449-31923-6.

[21] Xamarin Documentation. *Microsoft* [online]. Microsoft, ©2018 [cit. 2018-05-11]. Dostupné z: https://docs.microsoft.com/en-gb/xamarin/

[22] Apple Developer. *Apple, Inc* [online]. Apple, Inc, ©2018 [cit. 2018-05-11]. Dostupné z: https://developer.apple.com/develop/

[23] FINDER TEAM. iOS: Everything you need to know about Apple's mobile OS. In: *Finder.com* [online]. 19. 2. 2018 [cit. 2018-05-11]. Dostupné z: https://www.finder.com/ios-operating-system

[24] Smartphone OS. *IDC* [online]. IDC, ©2018 [cit. 2018-05-11]. Dostupné z: https://www.idc.com/promo/smartphone-market-share/os

[25] LUTZ, Mark. *Learning Python*. Fifth edition. Beijing: O'Reilly, 2013. ISBN 978-1449355739.

[26] Hashovací funkce. *Mendelova univerzita v Brně* [online]. Brno: Mendelova univerzita v Brně, © 2018 [cit. 2018-05-11]. Dostupné z: https://is.mendelu.cz/eknihovna/opory/zobraz_cast.pl?cast=7029

[27] What is an API? (Application Programing Interface). *MuleSoft, Inc* [online]. MuleSoft, Inc, ©2018 [cit. 2018-05-11]. Dostupné z: https://www.mulesoft.com/resources/api/what-is-an-api

[28] KORÁB, Vojtěch, Mária REŽŇÁKOVÁ a Jiří PETERKA. *Podnikatelský plán*. Brno: Computer Press, 2007. Praxe podnikatele. ISBN 978-80-251-1605-0.

[29] Exponea Guide. *Exponea* [online]. Exponea, ©2018 [cit. 2018-05-11]. Dostupné z: https://guides.exponea.com/

[30] Amazon Redshift Documentation. *Amazon Web Services* [online]. Amazon Web Services, ©2018 [cit. 2018-05-11]. Dostupné z: https://aws.amazon.com/documentation/redshift/

[31] Google BigQuery Documentation. *Google Cloud* [online]. Google Developers, ©2018 [cit. 2018-05-11]. Dostupné z: https://cloud.google.com/bigquery/docs/

[32] SQL Data Warehouse Documentation. *Microsoft Azure* [online]. Microsoft, ©2018 [cit. 2018-05-11]. Dostupné z: https://docs.microsoft.com/en-us/azure/sql-data-warehouse/

[33] Snowflake Documentation. *Snowflake Computing Inc* [online]. Snowflake Computing Inc, ©2018 [cit. 2018-05-11]. Dostupné z: https://docs.snowflake.net/manuals/index.html

[34] MongoDB Atlas. *MongoDB, Inc* [online]. MongoDB Inc, ©2018 [cit. 2018-05-11]. Dostupné z: https://docs.atlas.mongodb.com/?_ga=2.89040667.1208803040.1526046362-1746865972.1525876141

[35] Apache Drill Documentation. *The Apache Software Foundation* [online]. The Apache Software Foundation, ©2014 [cit. 2018-05-11]. Dostupné z: http://drill.apache.org/docs

[36] Impala Guide. *Cloudera, Inc* [online]. Cloudera, Inc, ©2014 [cit. 2018-05-11]. Dostupné z: http://drill.apache.org/docs

[37] Alooma Docs. *Alooma, Inc* [online]. Alooma, Inc, ©2018 [cit. 2018-05-11]. Dostupné z: https://support.alooma.com/hc/en-us

[38] Documentation. *Segment* [online]. Segment, ©2018 [cit. 2018-05-11]. Dostupné z: https://segment.com/docs/

Seznam tabulek

Seznam obrázků

Seznam grafů