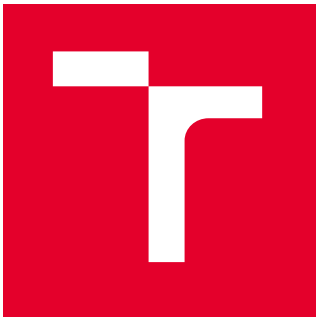



# VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

## FAKULTA PODNIKATELSKÁ

FACULTY OF BUSINESS AND MANAGEMENT

## ÚSTAV INFORMATIKY

INSTITUTE OF INFORMATICS


# NÁVRH A IMPLEMENTACE ŘEŠENÍ BUSINESS INTELLIGENCE V MALÉM PODNIKU

DESIGN AND IMPLEMENTATION OF BUSINESS INTELLIGENCE SOLUTIONS IN A SMALL BUSINESS

**DIPLOMOVÁ PRÁCE**
MASTER'S THESIS


**AUTOR PRÁCE** — **Bc. Gergő Viskievič**
AUTHOR

**VEDOUCÍ PRÁCE** — **Ing. Jan Luhan, Ph.D., MSc**
SUPERVISOR



**BRNO 2023**

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA PODNIKATELSKÁ

# Zadání diplomové práce

| | |
|---|---|
| Ústav: | Ústav informatiky |
| Student: | **Bc. Gergő Viskievič** |
| Vedoucí práce: | **Ing. Jan Luhan, Ph.D., MSc** |
| Akademický rok: | 2022/23 |
| Studijní program: | Informační management |

Garant studijního programu 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 a implementace řešení Business Intelligence v malém podniku

**Charakteristika problematiky úkolu:**

Úvod
Cíle práce, metody a postupy zpracování
Teoretická východiska práce
Analýza současného stavu
Vlastní návrhy řešení
Závěr
Seznam použité literatury
Přílohy

**Cíle, kterých má být dosaženo:**

Cílem práce je navrhnout řešení z oblasti Business Intelligence pro konkrétní subjekt se zaměřením do oblasti návrhu a implementace datového skladu, jako jednotného zdroje pravdy pro efektivní práci s daty a následnou podporu rozhodování.

**Základní literární prameny:**

GROSSMANN, W. and S. RINDERLE-MA. Fundamentals of Business Intelligence. 1st ed. Heidelberg: Springer Berlin / Heidelberg, 2015. 366 p. ISBN 978-3-662-46530-1.

LACKO, L. Databáze: datové sklady, OLAP a dolování dat s příklady v Microsoft SQL Serveru a Oracle. 1. vyd. Brno: Computer Press, 2003. 486 s. ISBN 80-7226-969-0.

NOVOTNÝ, O., J. POUR a D. SLÁNSKÝ. Business Intelligence: jak využít bohatství ve vašich datech. 1. vyd. Praha: Grada, 2005. 256 s. ISBN 80-247-1094-3.

SEIGE, V. Business intelligence: příručka manažera. 1. vyd. Praha: Tate International, 2007. 166s. ISBN 978-808-6813-127.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2022/23

V Brně dne 5.2.2023

L. S.

| | |
|---|---|
| doc. Ing. Miloš Koch, CSc.<br>garant | doc. Ing. Vojtěch Bartoš, Ph.D.<br>děkan |

**Abstrakt**

Diplomová práca sa zaoberá návrhom a implementáciou Business Intelligence riešenia so zameraním do oblasti návrhu a implementácie dátového skladu a reportingu v malom podniku, podnikajúci v oblasti digitálnych služieb. Predstavuje oblasť Business Intelligence, spôsoby implementácie a technologické riešenia. Na základe analýzy informačných potrieb podniku a používaných systémov je navrhnutá a implementovaná kompletná architektúra Business Intelligence.

**Kľúčové slová**

Business Intelligence, dátový sklad, ETL proces, budovanie dátového skladu, reporting, implementácia, riešenie, malý podnik

**Abstract**

The diploma thesis focuses on the design and implementation of a Business Intelligence solution with a dedication to the design and implementation of a data warehouse and reporting in a small business, operating in the field of digital services. It introduces the field of Business Intelligence, implementation methods and technology solutions. Based on the analysis of information needs of the business and used systems, a complete Business Intelligence architecture is designed and implemented.

**Keywords**

Business Intelligence, data warehouse, ETL process, data warehouse building, reporting, implementation, solution, small business

**Bibliografická citácia**

VISKIEVIČ, Gergő. Návrh a implementace řešení Business Intelligence v malém podniku [online]. Brno, 2023 [cit. 2023-05-15]. Dostupné z: https://www.vut.cz/studenti/zav-prace/detail/151933. Diplomová práce. Vysoké učení technické v Brně, Fakulta podnikatelská, Ústav informatiky. Vedoucí práce Jan Luhan.

**Čestné prehlásenie**

Prehlasujem, že predložená diplomová práca je pôvodná a spracoval som ju samostatne. Prehlasuje, že citácie použitých prameňov je úplná, že som v práci neporušil autorské práva (v zmysle Zákona č. 121/2000 Sb., o práve autorskom a o právach súvisiacich s právom autorským).

V Brne dňa 14. mája 2023

……..

podpis autora

**Poďakovanie**

Rád by som poďakoval vedúcemu mojej práce Ing. Janovi Luhanovi, Ph.D., MSc za odborné vedenie a cenné rady, ktoré mi poskytol.

Ďalej by som sa rád poďakoval Ing. Pavlovi Šabatkovi za vecné rady a pripomienky, ktoré mal k tejto diplomovej práci a za vypracovanie oponentúry tejto práce.

Taktiež ďakujem firme House of Řezáč za možnosť realizácie tejto diplomovej práce.

V neposlednom rade poďakovanie patrí mojim najbližším, za podporu počas celého obdobia štúdia.

# Obsah

# Úvod

Každá podnikateľská činnosť a aktivita generuje veľké množstvo dát. V prostredí súčasného dynamického a konkurenčného sveta sa kladie čoraz väčší dôraz na efektívne využívanie týchto dát. Platí to pre veľké nadnárodné korporácie ale aj malé lokálne podniky, ktoré potrebujú udržať krok s konkurenčným prostredím a usmerňovať ich podnikateľské činnosti na základe skutočných a aktuálnych informácií.

Odpoveďou na tieto potreby je oblasť Business Intelligence, ktorá je zameraná na zber a konverziu surových dát do zmysluplných a akčných informácií. Tieto informácie sú strategickým zdrojom, pretože sú kľúčové vstupy pre manažérov, ktorí sa rozhodujú o budúcom vývoji organizácií. Transformácia dát na informácie musí byť včasná a výsledky musia byť interpretované čo najrýchlejšie, aby organizácie boli schopní ich ich využiť ako konkurenčné výhody, odhaliť príležitosti alebo reagovať na riziká.

Mnoho organizácií používa veľa rôznych informačných systémov, ktorých sú denne generované a ukladané veľké množstvo dát. Tieto dáta sú väčšinou uložené v rôznych formátoch pomocou odlišných technológií a v podnikoch neexistuje jednotný pohľad na svoje dáta. Interpretácia dát je často riešená manuálnymi a časovo neefektívnymi aktivitami, ktoré v záveru nemusia priniesť užitočné dopady. Využitie Business Intelligence je veľakrát spájané s veľkými korporáciami, avšak implementácia Business Intelligence nájde svoje využitie aj v malých a stredných podnikoch. Úskalím je, ako u každého informačného systému, správna implementácia a údržba.

Táto práca sa zaoberá návrhom a implementáciou riešenia Business Intelligence v malom českom podniku, pôsobiaci na trhu digitálnych služieb v oblasti UX/UI designu, webovej a dátovej analytiky, ktorý nevlastní žiadne Business Intelligence riešenie. Podnik plánuje dlhodobý rast, s čím mu riešenie Business Intelligence môže rozsiahlo pomôcť v rozhodovacích procesoch.

# Ciele práce, metódy a postupy spracovania

Hlavným cieľom tejto diplomovej práce je návrh a implementácie riešenia Business Intelligence vo firme House of Řezáč s.r.o.

Výsledkom riešenia bude celopodnikový dátový sklad, ako jednotný zdroj pravdy, integrujúci podnikové dáta na jedno miesto, a postavený reporting na základe týchto dát. Cieľom tohto Business Intelligence riešenia je poskytnúť včasné a pravdivé informácie o výkone firmy vedeniu a podporiť jeho rozhodovacie procesy.

Prvá časť práce je venovaná teoretickým informáciám, ktoré sú potrebné k dosiahnutiu cieľa práce. Táto teoretická časť obsahuje komplexné informácie o oblasti Business Intelligence, jej využití v rôznych sférach podnikateľských činností a spôsoboch implementácie BI riešení. V ďalších častiach je predstavený proces dátového modelovania, ktorý je nezbytný pri implementácií dátového skladu a základné typy databáz pre vhodný výber technologických riešení. V závere teoretických východísk sa nachádzajú popisy internetovej komunikácie pomocou API, programovacieho jazyka Python a dvoch jeho knižníc, pre pochopenie a správnu implementáciu ETL procesov pomocou vlastných skriptov. Teoretická časť končí opisom jedným z nástrojov Business Intelligence, Looker Studio, ktorý bol jeden z hlavných požiadaviek firmy na riešenie BI.

Druhá časť práce sa zaoberá analýzou súčasného stavu firmy. Úvode je krátko predstavená firma, v ktorej bude riešenie BI implementovaná. Samotná analýza je poňatá v dvoch oblastiach. Prvá oblasť analýzy sa zaoberá informačnými potrebami a požiadavkami firmy na reporting. Druhá časť analýzy mapuje všetky využívané zdrojové systémy dát vo firme a spôsoby exportu dát z týchto systémov. Táto časť je kľúčová pre zistenie uskutočniteľnosti a splnenie informačných potrieb firmy.

Tretia časť práce obsahuje vlastné návrhy a spôsob implementácie riešenia BI od ETL procesov, cez budovanie dátových skladov, až po návrh reportingu v nástroji Looker Studio. Pre existenciu fungujúcej IT štruktúry firmy v podobe interného NAS serveru je väčšina riešení vlastná a implementovaná pomocou vlastných vyvinutých komponent

BI. V závere práce sú navrhované riešenie časovo ohodnotené, predstavené prínosy riešenia pre prácu a predstavené možné budúce smerovanie riešenia.

# 1 Teoretické východiská práce

Teoretické východiská práce sú štruktúrované tak, aby predstavili a komplexne rozobrali riešenú oblasť Business Intelligence, predstavili všeobecné a konkrétne použité techniky a technológie v budovaní dátového skladu a reportingu. Na základe znalostí z teoretických východísk vychádzajú navrhované a implementované riešenia BI pre firmu.

## 1.1 Business Intelligence

Business Intelligence (BI) predstavuje súbor metód, procesov, technológií a nástrojov, ktoré umožňujú zhromažďovať, spracovávať, analyzovať a vizualizovať dáta s cieľom získavania hodnotných informácií a podporovania efektívnych rozhodovaní v organizácii. Hlavným cieľom BI je transformovať surové dáta z rôznych zdrojov na zmysluplné a prístupné informácie, ktoré poskytujú podklad pre manažérov a zamestnancov pri tvorbe strategických a operatívnych rozhodnutí. [1]

Business Intelligence je dôležitým faktorom v moderných podnikoch, pretože podporuje zlepšovanie výkonnosti, zvyšovanie konkurencieschopnosti, identifikáciu príležitostí a rizík a optimalizáciu procesov. Okrem toho Business Intelligence umožňuje organizáciám lepšie pochopiť svojich zákazníkov, monitorovať trhové trendy a analyzovať finančné a operatívne výsledky.

*„Business Intelligence je sada procesů, aplikací a technologií, jejichž cílem je účinně a účelně podporovat rozhodovací procesy ve firmě. Podporují analytické a plánovací činnosti podniků a organizací a jsou postaveny na principech multidimenzionálních pohledů na podniková data.“ [3]*

Čerpané informácie v kapitole Business Intelligence pochádzajú zo zdrojov [1, 2, 3, 4].

### 1.1.1 Dáta, informácie, znalosti, múdrosť

V kontexte Business Intelligence je potrebné si uvedomiť a rozlíšiť medzi pojmami dáta, informácie, znalosti a múdrosť, pretože tieto pojmy predstavujú rôzne úrovne spracovania a využívania údajov v organizácií. Táto podkapitola čerpá informácie z [5].

**Dáta**

Dáta sú surové, neupravené, neštruktúrované jednotky prezentované faktami, číslami, textami alebo meraniami. Nie sú nijako organizované a neposkytujú žiadne ďalšie informácie týkajúce sa obsahu. Dáta majú zvyčajne najmenší dopad na manažerské rozhodovania v organizácií. V Business Intelligence predstavujú základný stavebný kameň, z ktorých sa vytvárajú informácie. Dáta sú vytvárané a pochádzajú z rôznych zdrojov dát, ako sú databázy, webové služby, CRM a ERP systémy a ďalšie.

**Informácie**

Informácie sú spracované a organizované dáta, ktoré nesú kontext a význam. Aby informácia vznikla, je potrebné získať kontext dát, kategorizovať ich, spočítať alebo agregovať. Zvyčajne odpovedajú na otázky ako “kto?” “čo?” “kde?” “kedy?” “koľko?”. Informácie umožňujú manažérom a zamestnancom lepšie pochopiť situáciu, identifikovať vzory a trendy a pomáhať pri uskutočňovaní efektívnych rozhodnutí. V rámci Business Intelligence úlohu konvertovania dát na informácie majú procesy extrakcie, transformácie a načítania (ETL), analýzy a vizualizácie.

**Znalosti**

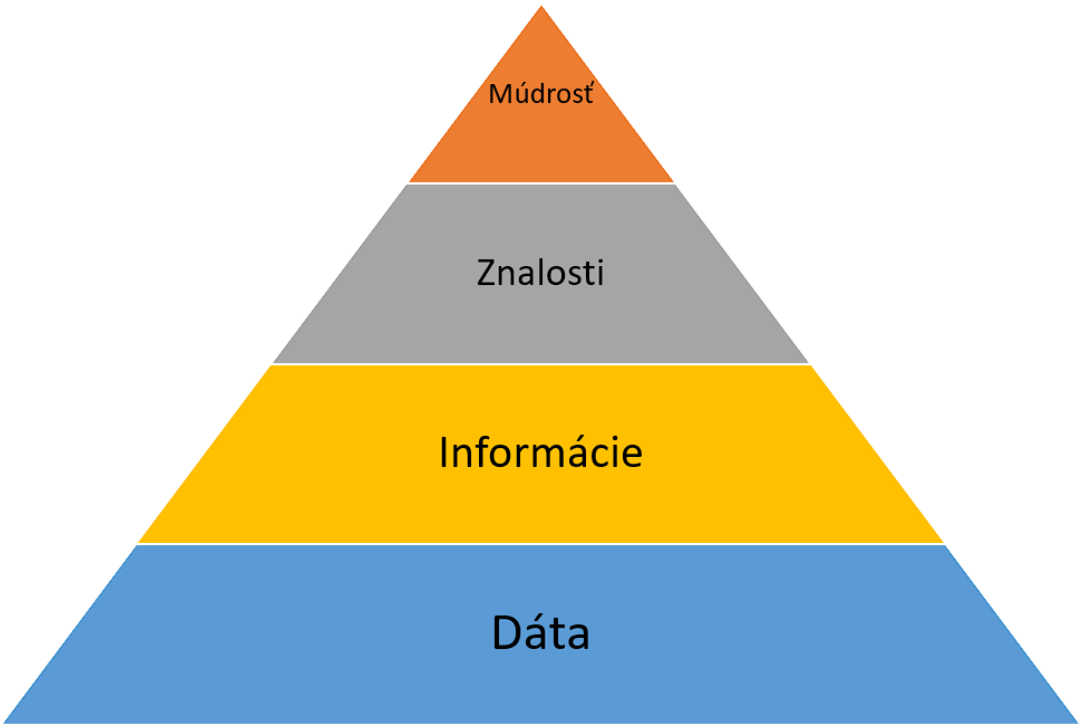
Znalosti sú výsledkom analýzy a interpretácie informácií. Zahrňujú know-how, porozumenie informácií, skúsenosti, pravidlá, expertné heuristiky, na základe ktorých sa vyhodnocujú podnety z okolia. Súčasťou znalostí je aj porozumenie vzťahov medzi rôznymi informáciami a schopnosť využiť tieto vzťahy na riešenie problémov alebo pretransformovať znalosti na konkurenčnú výhodu. Znalosti zvyčajne odpovedajú na otázku “ako?”. V Business Intelligence predstavujú znalosti schopnosť organizácie

používať informácie na základe skúseností, expertízy a porozumenia k lepšiemu rozhodovaniu, predikcií a plánovaniu.

**Múdrosť**

Múdrosť predstavuje najvyššiu úroveň v hierarchickom reťazci dáta-informácie-znalosti-múdrosť. Múdrosťou sa chápe schopnosť použiť znalosti v správnom čase a kontexte pre efektívne rozhodovanie. V Business Intelligence múdrosť predstavuje schopnosť manažérov a zamestnancov aplikovať svoje znalosti a skúsenosti pri riešení komplexných problémov, reagovať na situácie a prispôsobiť sa dynamickým trhovým podmienkam.

Cieľom Business Intelligence je transformovať dáta na informácie, získavať znalosti z týchto informácií a nakoniec dosiahnuť múdrosť prostredníctvom aplikácie týchto znalostí. Takýto proces pomáha organizáciám vytvárať hodnotu, optimalizovať svoje procesy a zlepšovať konkurencieschopnosť na trhu.




**Obrázok č.1:** Pyramída: dáta, informácie, znalosti, múdrosť [5] (Vlastné spracovanie)

### 1.1.2 Aplikačné oblasti Business Intelligence

Business Intelligence je možné využiť v každej oblasti podnikových činností a štruktúr. Základným predpokladom BI je generovanie a existencia dát, a potreby pochopenia, práce a vyhodnotenia týchto dát. Nasledujúce podkapitoly čerpajú informácie z [3].

#### 1.1.2.1 Financie

Využitím Business Intelligence je možné dostať pod kontrolu financie organizácie. Pomocou dát o uskutočnených finančných a účtovníckych operácií a ich uložením do dátového skladu, je možné získať hodnoty ukazovateľov o finančnej výkonnosti celej organizácie a jej jednotlivých oddelení a súčastí. Nasadenie riešení Business Intelligence v organizáciách zvyčajne prinesie finančnú transparentnosť v oblastiach riadenia nákladov. Okrem toho, Business Intelligence pomáha organizáciám:

- finančne plánovať a prognózovať,
- konsolidovať financie naprieč celú organizáciu,
- analyzovať náklady a ziskovosť spojené s produktami, dodávateľmi, marketingovými kanálmi, projektmi,
- optimalizovať financie.

#### 1.1.2.2 Marketing

Business Intelligence je jednou z integrálnych súčastí marketingových systémov typu CRM (Customer Relationship management) a Customer Intelligence. Naviac, BI aplikácie je možné použiť v oblastiach:

- riadenie portfólia produktov a služieb z hľadiska profitability a nákladovosti,
- klasifikácia a segmentácia zákazníkov,
- plánovanie a analýza dopadov marketingových kampaní,
- analýza marketingových nákladov a zdrojov,
- vyhodnotenie kampaní.

#### 1.1.2.3 Výroba

Aplikácie Business Intelligence vo výrobe sa používajú hlavne v oblasti riadenia kvality vo výrobnom procese, ale môžu sa nájsť aj nasledujúcich oblastiach:

- plánovanie a monitorovanie kľúčových ukazovateľov - doby dodávky, výroby, trvanie cyklov, priechodnosť výrobných línk,...
- analýza a plánovanie trendov na základe historických dát - simulácia výrobného procesu na základe dát,
- podpora nástrojov automatizovaného riadenia výrobného procesu - dodávanie informácií pre nápravu alebo zmenu výrobného procesu pre automatizované nástroje

#### 1.1.2.4 Logistika

Pomocou BI je možné evidovať a sledovať priebeh dodávok a vyhodnotiť tak celý proces dodávky produktov od dodávateľa k odberateľom. Umožňuje sledovať:

- efektívnosť dopravcov
- dopravné náklady
- kapacity
- doby dodávok
- dôvody vzniknutých problémov a reklamácií

#### 1.1.2.5 Dodávateľský reťazec

V rámci riadenia dodávateľov BI analyzuje a vyhodnocuje vzťahy s dodávateľmi, ich podmienky dodávok, ceny alebo kapacity. Riešenie BI je možné tiež použiť v prípadoch:

- analýza nákupu
- hodnotenie a výber dodávateľov
- podpora stratégie nákupu

#### 1.1.2.6 Ľudské zdroje

V tejto oblasti existujú špecializované aplikácie, ktoré sú využívané hlavne veľkými podnikmi. Slúžia na sledovanie stratégie organizácie v tejto oblasti v rôznych úrovniach organizačných jednotiek alebo zamestnancov. BI je používaný v oblastiach:

- analýza pracovnej sily
- analýza nákladov pracovnej sily
- motivácia zamestnancov

#### 1.1.2.7 Informatika

V oblasti informatiky sa nachádza veľký počet ukazovateľov, ktoré sledujú efektivity a výkony informačných systémov. BI umožňuje znižovať náklady, zefektívniť využívanie a maximalizovať výnosy informačných systémov. V riadení informatiky sa používa BI v oblastiach:

- riadenie bezpečnosti a rizík
- sledovanie a analýza poskytovaných služieb
- sledovanie a analýza zdrojov informačných systémov

#### 1.1.2.8 Riadenie výkonu podniku

Corporate Performance Management (Riadenie výkonu podniku) je oblasť, ktorá sa zaoberá komplexnými metódami, procesmi, ukazovateľmi a systémami na monitorovanie a riadenie výkonnosti a stratégie organizácií. Činnosti, s ktorými pracuje, sú:

- tvorba podnikovej stratégie
- nastavovanie cieľových hodnôt ukazovateľov na rôznych úrovniach podnikania
- monitorovanie plnenia cieľov
- vytváranie predpovedí a plánov na budúce obdobia

#### 1.1.2.9 Webová analytika

Webová analytika je odbor, ktorá sa zaoberá dátmi o chovaní užívateľov na webových stránkach. Aplikácie Business Intelligence vo webovej analytike sú špecializované pre zber a analýzu dát webových aplikácií. Tieto aplikácie poskytujú nasledujúce formy analýz:

- celkové ukazovatele návštevnosti - statistické informácie o počet návštev podľa jednotlivých stránok, delený podľa času, typu zariadenia,...
- chovanie návštevníkov - analýzy, ktoré sa zaoberajú s komplexnými vzormi chovania návštevníkov, za účelom zmien a optimalizácií webových stránok
- analýza marketingových kanálov - zaoberá sa nákladmi a výnosmi z provozu jednotlivých marketingových kanálov. Sledujú sa rôzne typy ukazovateľov od počtu akvizície nových užívateľov po celkovú hodnotu zákazníka.

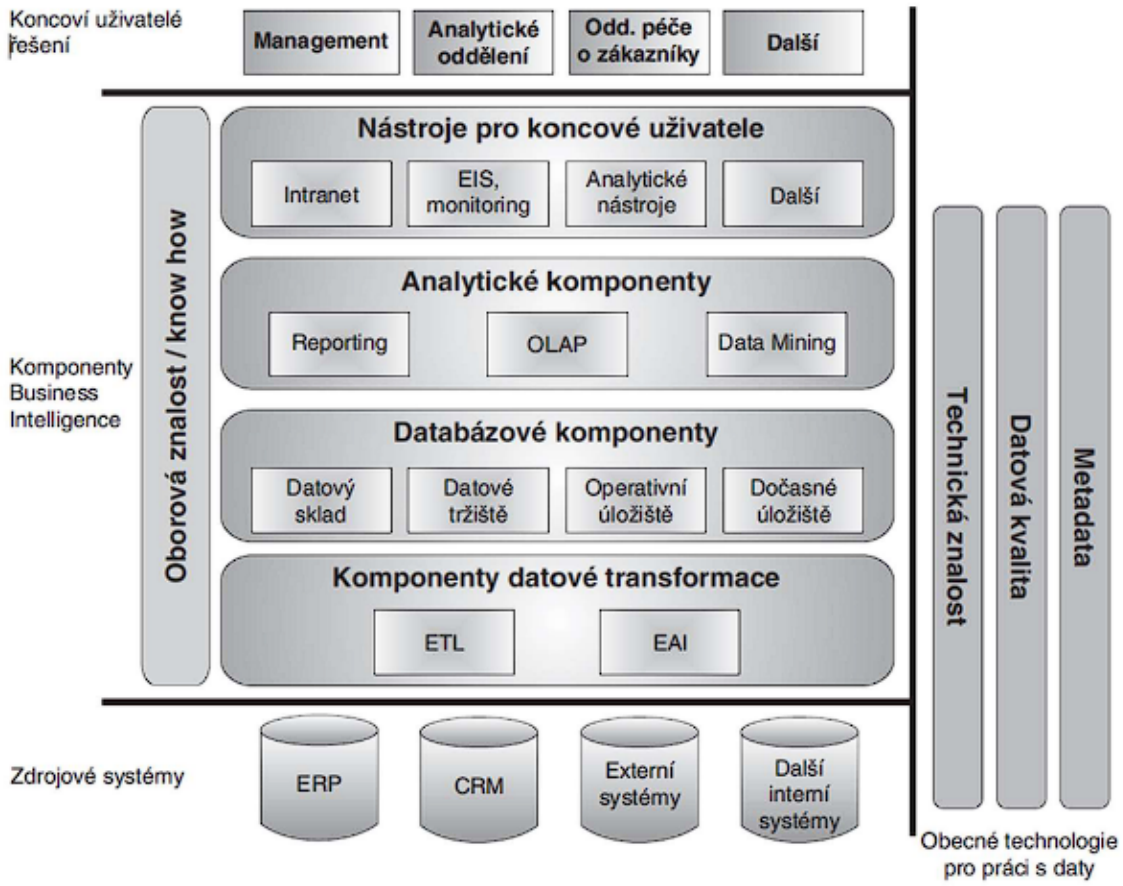
#### 1.1.2.10 Customer Intelligence

Customer Intelligence (CI) se súhrn informačných systémov zameraných na komplexné poznanie zákazníka. Tieto systémy zvyčajne kombinujú systémy BI a CRM. Cieľom CI je budovanie hlbokých a efektívnych vzťahov so zákazníkmi a následná podpora rozhodovania predajcov. Customer Intelligence spočíva v zbieraní základných dát o zákazníkoch a v následnom doplnení týchto dat o uskutočnených transakciách a ostatných interakciách so zákazníkom. Výsledkom toho je profil zákazníka s úplnou históriou jeho aktivít s organizáciou. Zdrojovými dátmi týchto aktivít môžu byť ERP systémy, marketingové aktivity v emailových a reklamných kampaniach, kontaktovania cez telefón a podobne. Na základe týchto dát je možné:

- automatizovať obchodné procesy,
- automatizovať marketingové procesy,
- poskytovať údaje pre podporu zákazníckych a servisných centier,
- segmentovať zákazníkov,
- analyzovať marketingové kampane,
- personalizovať ponuky zákazníkom.

### 1.1.4 Všeobecná architektúra Business Intelligence

Táto podkapitola sa zaoberá jednotlivými komponentmi, typmi aplikácií a vrstvami architektúry Business Intelligence. Pohľad na architektúru je zovšeobecnený, ale je potrebné si uvedomiť, že koncové usporiadanie jednotlivých komponentov v riešeniach Business Intelligence sa môžu veľmi líšiť od oblasti aplikácie, potrieb a typu organizácie, technologickej a finančnej komplexity riešenia. Táto podkapitola čerpá informácie z [3]. Nasledujúci obrázok znázorňuje všeobecnú architektúru Business Intelligence.




**Obrázok č. 2:** Všeobecná architektúra Business Intelligence (Zdroj: [3])

#### 1.1.4.1 Vrstvy Business Intelligence

Všeobecný koncept architektúry Business Intelligence obsahuje nasledujúce vrstvy [3]:

**Vrstva pre extrakciu, transformáciu, čistenie a nahrávanie dát**

Táto vrstva pokrýva zber, prenos, transformáciu a úpravu dát zo zdrojových aplikácií a systémov do vrstvy pre ukladanie dát v Business Intelligence. Táto vrstva pracuje v komponentami:

- ETL systémy - (Extract, Transform, Load) systémy pre extrakciu, transformáciu a nahrávania dát
- EAI systémy - (Enterprise Application Integration) systémy pre integráciu aplikácií

**Vrstva pre ukladanie dát**

Vrstva pre ukladanie dát má za úlohu ukladať, aktualizovať a spravovať dáta v riešeniach Business Intelligence. Patria sem komponenty:

- Datové sklady - základný databázový komponent riešenia BI
- Datové tržiská - subjektovo orientované analytické databázy
- Operatívne úložiská dát - podporné analytické databázy
- Dočasné úložiská dát - databázy slúžiace pre dočasné ukladanie dát pred ich spracovaním do databázových komponent

**Vrstva pre analýzu dát**

Vrstva pokrýva činnosti spojené s vyhodnotením, analýzou, interpretáciou a sprístupnením dát. Obsahuje komponenty:

- Reporting - analytická vrstva, ktorá sa zaoberá dotazovaniami do databázových komponent
- OLAP systémy - (Online Analytical Processing) systémy, ktoré sa zamerajú na komplexné a dynamické analytické úlohy
- Dolovanie dát - systémy a algoritmy pre získavanie znalostí z databáz

**Prezentačná vrstva**

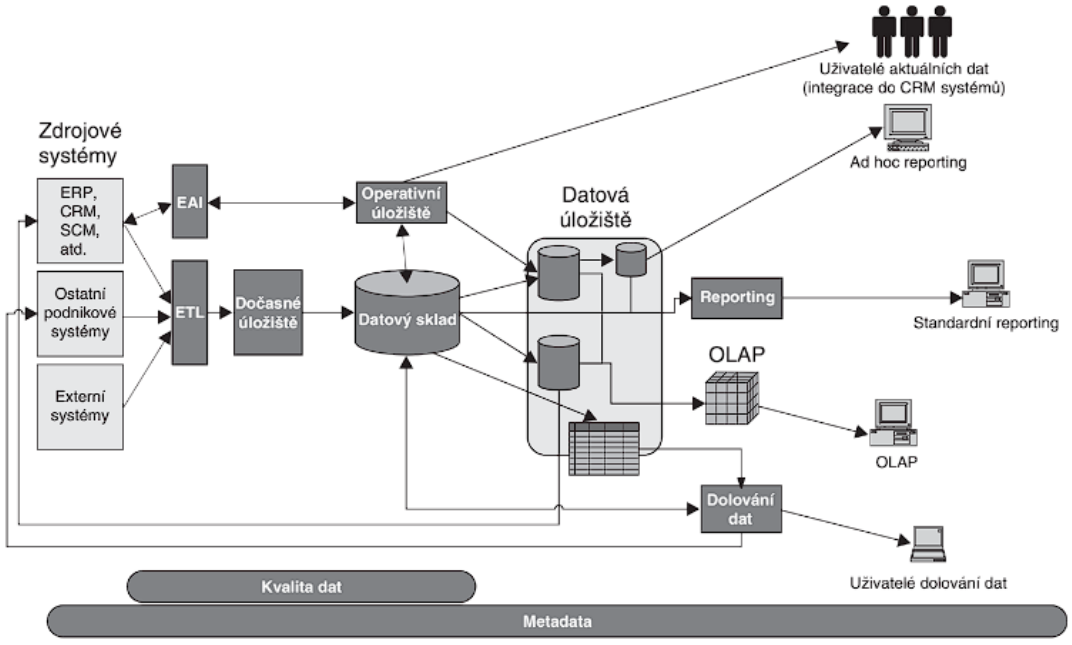
Zahŕňa nástroje a aplikácie používané koncovými užívateľmi na komunikáciu s ostatnými komponentami BI. Patria sem rôzne portálové WWW aplikácie, analytické aplikácie, atď.

**Vrstva odbornej znalosti**

Zahŕňa odborné znalosti a best-practices v nasadzovaní riešení Business Intelligence pre konkrétny účel.

### 1.1.4.2 Komponenty Business Intelligence

V tejto podkapitole sú predstavené hlavné komponenty architektúry Business Intelligence podľa [3]. Nasledujúci obrázok zobrazuje hlavné komponenty BI a vzťahy medzi nimi.




**Obrázok č. 3**: Komponenty BI a vzťahy medzi nimi (Zdroj: [3])

### Produkčné systémy

Produkčné, často označované ako zdrojové, primárne, transakćné alebo OLTP, systémy sú také systémy, z ktorých komponenty BI získavajú dáta. Tieto systémy väčšinou predstavujú aplikácie alebo softvéry, ktoré sú primárne používané zamestnancami pri plnení ich práce. Podporujú ukladanie a modifikáciu dát v reálnom čase a nie sú navrhnuté pre analytické úlohy. Takýmito systémami môžu byť ERP, CRM, SCM systémy, systémy využívané vo finančných, personálnych oddeleniach organizácií. Produkčné systémy predstavujú hlavný zdroj dát BI. Väčšinou sa jedná o niekoľko rôznych produkčných systémov, v ktorých sú dáta uložené obsahovo a technologicky odlišne.

### Extraction, Transformation, Loading - ETL

ETL systémy, často označované ako dátové pumpy, sú jedným z najdôležitejších komponent BI. Jeho úlohou je dostať dáta zo zdrojových systémov, vyčistiť a upraviť ich do požadovanej štruktúry a následne ich nahrať do požadovaných štruktúr a schémat dátového skladu. Používajú sa na prenos dát medzi dvoma a viacerými systémami. ETL nástroje pracujú v dávkovom režime, čo znamená, že dáta sú prenášané v určitých časových intervaloch - denné, týždenné, mesačne intervaly. Avšak existujú aj ETL nástroje, ktoré sú schopné dáta prenášať v reálnom čase. Takýto režim sa volá *streaming*.

### Enterprise Application Integration

Tieto systémy sa používajú vo vrstve zdrojových systémov. Ich cieľom je integrovať podnikové systémy a redukovať používané rozhrania zamestnancami. Nie je nutným komponentom riešení BI. Používajú sa na dvoch úrovniach:

- úroveň dátovej integrácie - EAI nástroje sú použité na integráciu a distribúciu dát
- úroveň aplikačnej integrácie - EAI nástroje sú naviac použité aj na zdieľanie vybraných funkcií informačných systémov

### Dočasné úložisko dát

Dočasné úložisko dát - DSA (Data Staging Areas) je miesto, kam sa dočasne ukladajú extrahované dáta z produkčných systémov pre rýchlu a efektívnu extrakciu. Zvyčajne sa ukladajú netransformované dáta. Je to nepovinný komponent riešení BI, ktorý sa používa pri:

- zaťažených produkčných systémoch, kde je potrebné previesť ETL procesy s minimálnym dopadom na výkonnosť systémov
- systémoch, kde je potrebné dáta pred spracovaním konvertovať do databázového formátu

### Operatívne úložisko dát

Operatívne úložisko dát - ODS (Operational Data Store) je nenutným komponentom BI, ktorý môže mať dva podoby:

- Operatívne úložisko dát funguje ako jednotné miesto dátovej integrácie časovo aktuálnych dát z primárnych systémov. Zvyčajne obsahujú konsolidované agregované dáta s minimálnou dobou odozvy. Veľakrát sú tieto úložiská napojené na EAI a podporujú tak vkladanie a modifikáciou dát v reálnom čase.
- Funguje ako databáza podporujúca relatívne jednoduché dotazy nad malým množstvom aktuálnych analytických dát. Vzniká ako derivácia dátového skladu a obsahuje len podmnožinu dát.

### Dátový sklad

Dátový sklad - DWH (Data warehouse) je špecializovaná databáza, ktorá slúži na ukladanie, spracovanie a analýzu veľkého množstva historických dát z rôznych zdrojov. Datové sklady sú navrhnuté tak, aby zvládali komplexné dotazy a agregácie dát, čím umožňujú získavanie presných a konzistentných informácií. Dátové sklady majú nasledujúce vlastnosti:

- subjektovo orientovaný - dáta sú uložené podľa typu dát a nie podľa aplikácií, v ktorých vznikla. Podporuje to jednotný pohľad na jeden subjekt, napr. zamestnanca.
- integrovaný - dáta sú ukladané z celej organizácie.

- stály - dáta sú načítané z operatívnych databáz, v priebehu času sa nemenia a existujú po celú dobu života dátového skladu.
- časovo rozlíšený - dátové sklady obsahujú historické dáta a každý záznam nesie informáciu o dimenzií času.

**Dátové tržisko**

Dátové tržisko - DMA (Data Mart) funguje podobne ako dátové sklady. Rozdiel je v tom, že v princípe dátové tržiská sú určené pre obmedzený okruh užívateľov (podľa oddelenia organizácie, pobočky, závodu, krajiny). Sú to decentralizované dátové sklady, ktoré sú eventuálne integrované do celopodnikového dátového skladu. V určitých prípadoch môžu dátové tržiská slúžiť ako medzistupeň pri transformácií dát z produkčných databáz.

**OLAP databáze**

OLAP databázy obsahujú jednu alebo viac vzájomne prepojených OLAP kociek. Kocky OLAP na rozdiel od dátových skladov obsahujú vopred spracované agregácie dát podľa jasne definovaných hierarchických štruktúr dimenzií. Technológia OLAP môže byť realizovaná v nasledujúcich variantoch:

- MOLAP (Multidimensional OLAP) - uloženie dát v multidimenzionálnych binárnych OLAp kockách.
- ROLAP (Relational OLAP) - multidimenzionalita je riešená uložením dát v relačných databázach.
- HOLAP (Hybrid OLAP) - je kombinácia predchádzajúcich riešení. Dáta sú uložené v relačných databázach a agregované hodnoty sú uložené v binárnych OLAP kockách.

**Reporting**

Reporting predstavuje činnosti spojené s dotazovaním sa do databáz pomocou rozhraní databáz (napr. SQL dotazy). Reporting môže byť:

- štandardný reporting, ktorý je uskutočnený v určitých časových periódach s predpripravenými dotazmi.

- ad-hoc reporting, ktorý zahrňuje jednorázové a špecifické dotazy užívateľom mimo štandardný reporting.

Podkapitola 1.1.8 sa ďalej venuje téme Reportingu.

**Data mining**

Väčšina organizácií produkuje veľké množstvo dát, ktoré môžu obsahovať ukryté vzorce chovania a zásadné informácie pri konkurenčnom boji. Data mining, dolovanie znalostí z databáz, je proces extrakcie relevantných, predom neznámych a nedefinovaných informácií z existujúcich dát. Tieto procesy majú podobu špeciálnych algoritmov, ktoré používajú štatistické a matematické techniky. Medzi postupy dolovania dát patria neurónové siete, rozhodovacie stromy, genetické algoritmy, klasifikace, clustering, asociaćné pravidlá, sekvenčné vzory. Viac informácií o data mining algoritmoch sa nachádza v zdrojoch [1,3].

**Nástroje pre správu kvality dát**

Pre analytické účely je podstatné, aby dáta boli správne a zobrazovali skutočnú situáciu podniku. Preto sú nástroje na zabezpečenie kvality dát dôležitou súčasťou každého BI riešenia, ktoré zabezpečujú dosahovanie požadovaných vlastností dát. Kvalita dát nie je problémom výlučne riešení BI, ale týka sa všetkých zdrojov dát podniku. V zásade platí, že kvalita výstupov môže byť len taká dobrá, ako kvalita vstupov. Existuje mnoho vlastností, ktoré sú možné sledovať pri dátovej kvalite. Najpoužívanejšie sú:

- Úplnosť - Táto vlastnosť sa týka toho, či sú dáta kompletné a obsahujú všetky potrebné informácie pre účely analýzy. Úplnosť zahŕňa kontrolu chýbajúcich alebo neúplných údajov, ktoré by mohli ovplyvniť kvalitu analýzy.
- Súlad - Súlad sa týka zhody dát s preddefinovanými štandardmi, pravidlami alebo formátmi. Napríklad, či sú dáta zosúladené v rámci jednotiek, dátumov alebo časových zón. Súlad zabezpečuje, že dáta sú konzistentné a porovnateľné naprieč celou organizáciou.
- Konzistencia - Dáta sú navzájom konzistentné a nemajú žiadne rozpory. Táto vlastnosť sa týka udržiavania jednotnej štruktúry, formátu alebo hodnôt dát naprieč rôznymi zdrojmi a časovými úsekmi.

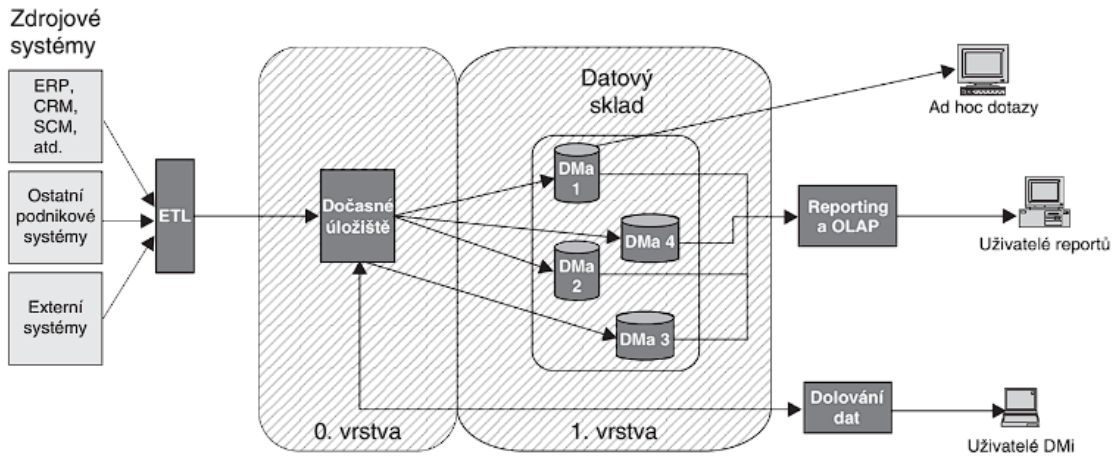
- Presnosť - Dáta správne a zodpovedajú skutočnosti. Presné dáta sú dôležité pre kvalitné rozhodovanie a analýzu, pretože nesprávne údaje môžu viesť k chybným záverom a rozhodnutiam.
- Unikátnosť - Unikátnosť sa týka zabezpečenia, že každá položka dát je jedinečná a neexistujú žiadne duplicity. Unikátnosť zabraňuje nejednoznačnosti a zlepšuje presnosť analýzy.
- Integrita - Týka udržiavania ich celistvosti a konzistentnosti v priebehu času a rôznych procesov. Táto vlastnosť zahŕňa kontrolu, či sú dáta nezmenené počas prenosu alebo ukladania.

### 1.1.5 Prístupy k budovaniu Business Intelligence

Kapitola 1.1.4 opisuje všeobecnú architektúru, vrstvy a komponenty riešení Business Intelligence. V praxi je však možné stretnúť sa s viacerými spôsobmi a prístupmi k riešeniu architektúry Business Intelligence a dátového skladu. Nasledujúce podkapitoly obsahujú opísané najvýznamnejšie spôsoby budovania BI a DWH [2,3].

#### 1.1.5.1 Postupné budovanie dátových tržísk

Tento koncept vytvoril R. Kimball, jeden z tvorcov budovania dátových skladov. Tento princíp spočíva v relatívne nezávislom vybudovaní jednotlivých dátových tržísk pre vybrané jednotky v organizácií (oddelenie, pobočka, krajina, závod). Takéto dátové tržisko je typicky plne životaschopné, obsahujúce všetky komponenty Business Intelligence podľa potrieb organizačnej jednotky - získavanie dát z produkčných systémov, spracovanie a ukladanie dát, reporting užívateľom.




**Obrázok č.4:** Architektúra postupného budovania dátových tržísk (Zdroj: [3])

**Kroky** postupného budovania dátových tržísk je nasledujúci:

- Na základe potrieb príslušnej organizačnej jednotky je vybudované prvé dátové tržisko. V rámci návrhu databázového modelu sú vytypované tie dimenzie, ktoré sa môžu zdieľať a používať naprieč jednotky a budúce dátové tržiská (čas, dimenzie zákazníka, dimenzie produktov).
- Ďaľsie dátové tržiská sú navrhované a budované tak, aby maximálne použili existujúce dimenzie z prvého dátového tržiska. Ak je potrebné, dimenzie sú upravené podľa potrieb ostatných dátových tržísk.
- Všetky ostatné komponenty v rámci BI riešenia sa budujú nezávisle od seba.

V prípade potreby je možne vytvoriť nad nezávislými dátovými tržiskami celopodnikovú dátovú vrstvu, ktorá umožnuje reporting dát naprieč tržiská.

**Výhody** takéhoto spôsobu budovania Business Intelligence sú nasledujúce:

- Analytické potreby jednotlivých organizačných jednotiek sú relatívne rýchlo uspokojené.
- Vybudovania jednotlivých tržísk sú riešené separátnymi projektmi na 3-6 mesiacov, čo umožňuje jednoduchší finanční controlling a sledovanie návratnosti investície.

**Nevýhody** postupného budovania BI sú:

- Vybudovanie jednotnej reportingovej vrstvy môže byť finančne a časovo náročné.

- Môžu existovať duplicitné komponenty.
- Pri potrebe dolovaní dat je potrebné vybudovať špeciálne riešenie mimo existujúce dátové tržiská.

Všeobecne **sa odporúča** použiť tento prístup v nasledujúcich situáciách:

- Nie je možné alebo nie je potrebné budovať celopodnikové riešenie.
- Je potrebné rýchlo vybudovať riešenie BI v danom oddelení, kde sa neočakáva budúca celopodniková integrácia.
- Nedostatok finančných prostriedkov na strane zadávateľa.

#### 1.1.5.2 Jednorázové vybudovanie celopodnikového riešenia

Koncept vytvorenia centrálneho celopodnikového dátového skladu a potrebných komponent BI je možné vytvoriť jednorázovým a prírastkovým spôsobom.

**Jednorázové riešenie**

Prístup spočíva v jednorázovom vybudovaní celopodnikového riešenia, ktoré pokrýva všetky potreby organizácie. **Kroky** sú nasledujúce:

- Analýza a dokumentácia všetkých užívateľských potrieb naprieč celej organizácie.
- Návrh a implementácia celkového riešenia - vybudovanie konsolidovaného dátového skladu a tvorba závislých dátových tržísk.
- V prípade nových užívateľských potrieb sú vytvorené nové dátové tržiská s využitím konsolidovaného dátového skladu.

**Výhody** takého prístupu sú:

- Architektúra je flexibilná na podporu náročných analytických úloh.
- Je možné budovať neobmedzené množstvo dátových tržísk podľa potreby.
- Reporting na základe celopodnikových dát.
- Komponenty BI riešení nie sú duplikované.

Medzi **nevýhody** prístupu patria:

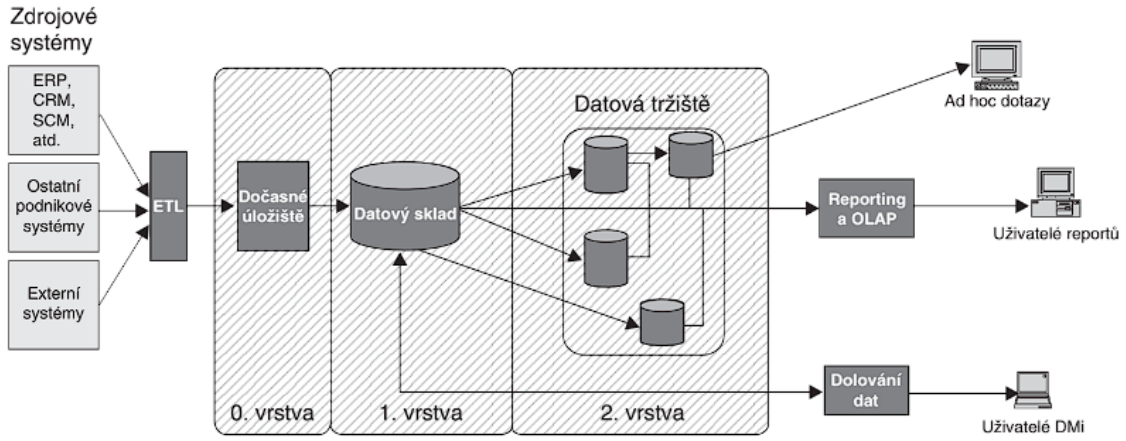
- Vývoj trvá dlho a stojí veľké množstvo finančných prostriedkov.
- Je obtiažne sledovať návratnosť investície.

- Riziko zmeny požiadaviek počas implementácie projektu.

Tento prístup je **vhodné** zvoliť v situáciách:

- Celkové riešenie je relatívne malé a je jednoduché zmapovať všetky užívateľské potreby.
- Neexistuje veľké riziko zmeny alebo rozšírenia užívateľských požiadaviek.

Architektúra jednorázového konsolidovaného dátového skladu sa nachádza na nasledujúcom obrázku.

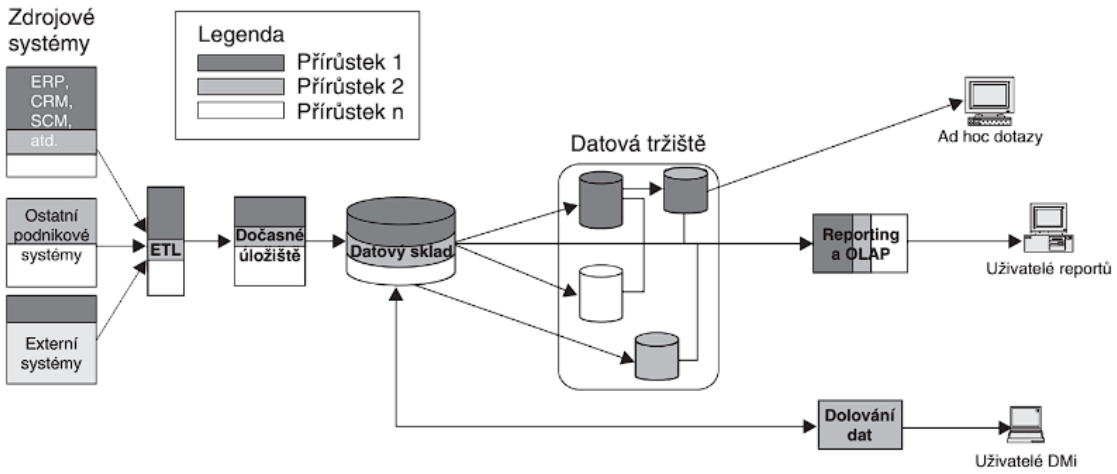



**Obrázok č. 5:** Architektúra jednorázového konsolidovaného dátového skladu (Zdroj: [3])

### Prírastkový prístup

Prírastkový prístup je spojený s architektúrou konsolidovaného dátového skladu a kombinuje výhody oboch predchádzajúcich spôsobov budovania dátových skladov. Priebeh tohto prístupu je nasledujúci:

- Vytvorí sa celopodnikový koncept Business Intelligence. Projekt obsahuje všetky užívateľské potreby, ich prioritizáciu, návrh architektúry riešení, identifikáciu jednotlivých prírastkových projektov a ich nadväznosť.
- Jednotlivé prírastkové projekty sú naplnené. Každý prírastok je kompletné riešenie BI, ktoré je otvorené a rozšíriteľné nadväzujúcim prírastkovým projektom.




**Obrázok č. 6:** Prírastkový prístup v riešení BI (Zdroj: [3])

**Výhody** tohto prístupu sú:

- Jednotlivé riešenia sú rýchlo implementované.
- Existuje priestor na reakciu zmien v potrebách organizácie.
- Je priestor na úpravy konceptu budovania BI.
- Jednotlivé prírastkové projekty sa dobre finančne sledujú.
- BI komponenty nie sú duplikované.

Medzi **nevýhody** patria:

- Je nutné investovať väčší čas do tvorby celopodnikového BI na začiatku prvého projektu.
- Relatívne neskoršie dodanie výsledkov prvého projektu z dôvodu tvorby celopodnikového konceptu.

Prístup je **vhodné** zvoliť ak:

- Je záujem o celopodnikové konsolidované riešenie ale očakáva sa rozvoj užívateľských potrieb.
- Je ochota investovať do vytvorenia celopodnikovej stratégie a následne budovania malých prírastkových riešení.

### 1.1.6 OLTP a OLAP

Komponenty Business Intelligence pracujú s dvoma typmi dát - operatívnymi (transakčnými) a analytickými. Oba tieto typy dát súvisia s rôznymi procesmi a technológiami využívané v Business Intelligence, ktoré umožňujú správu a analýzu týchto dát. Dve hlavné technológie používané v Business Intelligence sú OLTP a OLAP. Rozdiel medzi operatívnymi a analytickými dátami spočíva v účelu, pre ktorý sa využívajú, a v technológiách, ktorými sú spracované [2, 3].

**Operatívne (transakčné) dáta**

Tieto dáta sa týkajú bežných operácií a transakcií podnikateľských činností, ako sú napríklad predaje, nákupy, faktúry, platby, … Operatívne dáta sú obvykle ukladané v relačných databázach, ktoré sú optimalizované pre OLTP, nazývané transakčné databázy. OLTP (On-line Transaction Processing) databázy sú optimalizované a zamerajú sa na rýchle a efektívne spracovanie jednotlivých transakcií, ktoré môžu byť uskutočnené súčasne. Tieto databázy obsahujú detailné informácie o transakciách a sú využívané viacerými užívateľmi. Slúžia na vytváranie a modifikáciu dát v reálnom čase. Patria sem databázy systémov pre výdaj a príjem produktov, faktúr, účtovníckych operácií, atď. Nevýhoda týchto systémov je, že neboli vytvorené pre analýzu dát. Dotazovanie do týchto databáz je zvyčajne časovo náročná a môže dôjsť k zaťaženiu systémov.

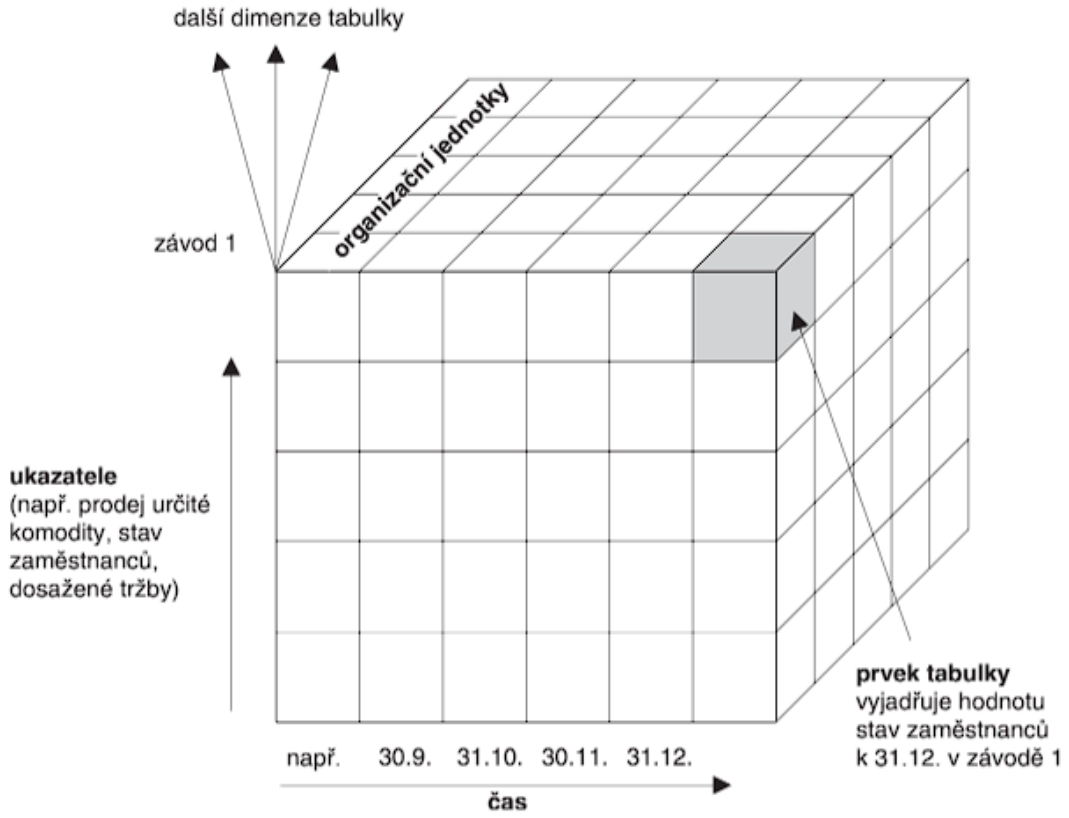
**Analytické dáta**

Analytické dáta sa vzťahujú k súhrnným, historickým a prediktívnym analýzam dát, ktoré umožňujú organizáciám získať vzhľad do svojich činností a podporovať efektívne rozhodovanie. Tieto dáta sú obvykle ukladané v databázach, ktoré sú optimalizované pre OLAP. OLAP (On-line Analytical Processing) sa zamerajú na efektívne spracovanie komplexných dotazov a analýz, ktoré potrebujú agregáciu a spracovanie veľkého množstva dát. OLAP predstavuje základný kameň dátových skladov a poskytuje prostredie pre efektívnu analýzu dát.

### 1.1.7 Multidimenzionalita

Multidimenzionalita je základný princíp Business Intelligence. Vyjadruje potrebu dívania sa na dáta z viacerých hľadísk, uhľov pohľadu a ich kombináciu. Analytické nástroje musia podporovať nachádzanie súvislostí, ktoré nie sú z dát jasné, prechádzať veľké množstvo dát, vypočítavať agregácie, meniť pohľady na dáta a ukladať ich do prehľadných tabuliek a grafov. Táto potreba sa rieši špeciálnou organizáciou dát v databázach - multidimenzionalitou [2,3].

Princípom multidimenzionality je niekoľkodimenzionálna tabuľka, ktorá umožnuje rychle meniť jednotlivé dimenzie a ponúkať tak rôzne pohľady na realitu.



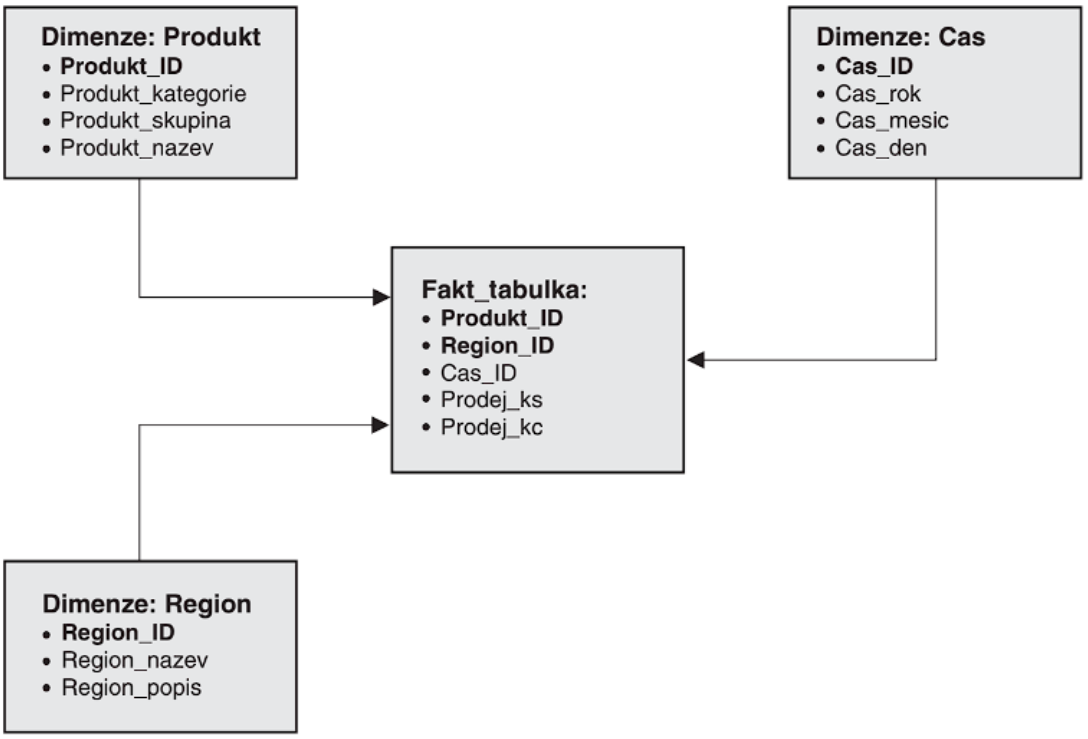

**Obrázok č.7:** Princíp multidimenzionálnej databázy (Zdroj: [3])

#### 1.1.7.1 Implementácia multidimenzionality v relačnej databáze

Relačný dimenzionálny model má dve základné podoby:

- schéma hviezdy (STAR scheme)
- schéma snehovej vločky (SNOWFLAKE scheme)

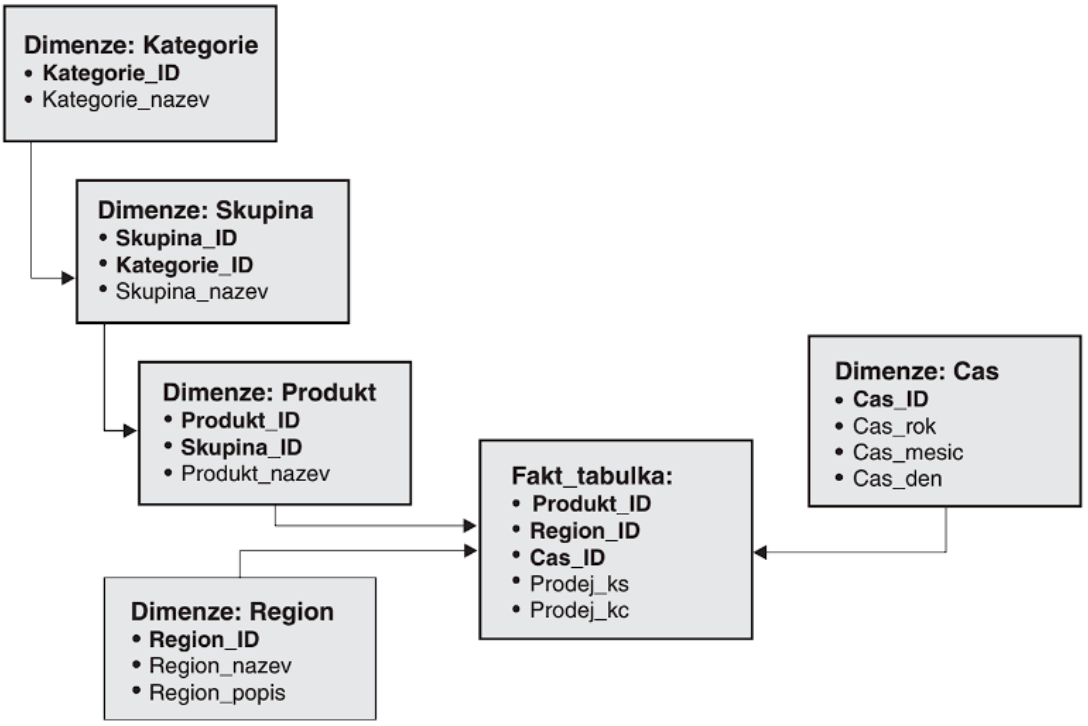
V centre oboch schém je tabuľka faktov, ktorá obsahuje sledované ukazovatele - môžu ním byť cena, marža, kusy. Ukazovatele sú identifikované cudzím kľúčom z dimenzionálnych tabuliek, v ktorých sú uložené jednotlivé hodnoty dimenzie. Dimenzionálne tabuľky je možné predstaviť ako číselník, ktorý obsahuje textové informácie o hodnotách uložených v tabuľke faktov. Do dimenzionálnych tabuliek sa ukladajú zvyčajne ukladajú textové, diskrétne, nemenné a nemerateľné hodnoty. Nasledujúci obrázok ukazuje schému hviezdy v dátovom modeli [2,3].




**Obrázok č. 8:** Schéma hviezdy v dátovom modele (Zdroj: [3])

V niektorých prípadoch je rozumné a potrebné niektoré dimenzionálne tabuľky ďalej normalizovať. Je to v prípade, keď jedna dimenzionálna tabuľka obsahuje celú hierarchickú štruktúru dimenzie, čím pri pridávaní alebo úprave záznamov v tabuľke sa

opakujú rovnaké hodnoty. Normalizáciou takýchto tabuliek sa vytvára tzv. schéma snehovej vločky, prezentovaná na nasledujúcom obrázku.




**Obrázok č. 9:** Schéma snehovej vločky v dátovom modele (Zdroj: [3])

### 1.1.8 Reporting a vizualizácia dát

Uložené dáta v podnikových systémoch nesú dôležité informácie o výkonnosti organizácie, zákazníkoch, dodávateľoch, zamestnancoch, produktoch a procesoch. Tieto dáta je potrebné vhodným spôsobom premeniť na informácie, interpretovať a vizualizovať príslušným užívateľom informácií. Pre tieto účely slúžia reporty, ktoré sú dokumenty štruktúrované tak, aby vhodným spôsobom interpretovali dáta a informácie pre jednoduché porozumenie, analýzu a podporu rozhodovania. Pomocou reportov je možné jednoducho sledovať dôležité ukazovatele podniku v čase a porovnávať ich s plánovanými hodnotami.

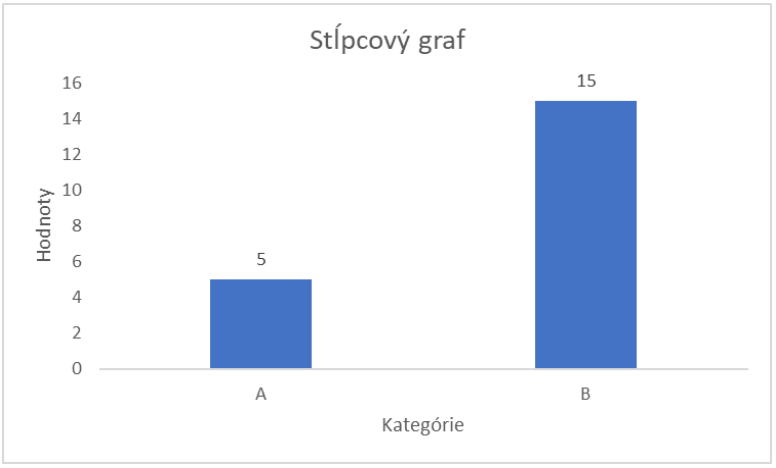
Dáta a informácie sú možné prezentovať mnohými formami. Je vždy dôležité vybrať vhodný vizualizačný prostriedok z niekoľkých dôvodov:

- Lepšie porozumenie dát: Správny vizualizačný nástroj uľahčuje rýchle pochopenie dát, odhaľuje trendy, vzorce a vzťahy, ktoré by bolo ťažké rozpoznať len prostredníctvom tabuliek alebo čísel.
- komunikácia informácií: Dobre navrhnutá vizualizácia a reporty dokážu rýchlo komunikovať a sprostredkovať kľúčové poznatky a podporiť rozhodovanie
- Udržanie pozornosti: Vhodný vizualizačný nástroj môže udržať pozornosť užívateľa, uľahčiť zapamätanie informácií a zvýšiť záujem.
- Prispôsobenie užívateľom: Rôzny užívatelia môžu mať rôzne preferencie a úrovne znalostí v oblasti vizualizácií. Správny výber vizualizačného nástroja pomáha zabezpečiť, aby boli informácie zrozumiteľné a prístupné pre všetkých užívateľov.

Nasledujúci zoznam prezentovaných typov vizualizácií nie je konečný. Rôzne typy vizualizácií a vizualizačných techník sa nachádza v [1], z ktorého čerpá táto kapitola a jej podkapitola informácie.

#### 1.1.8.1 Stĺpcový graf

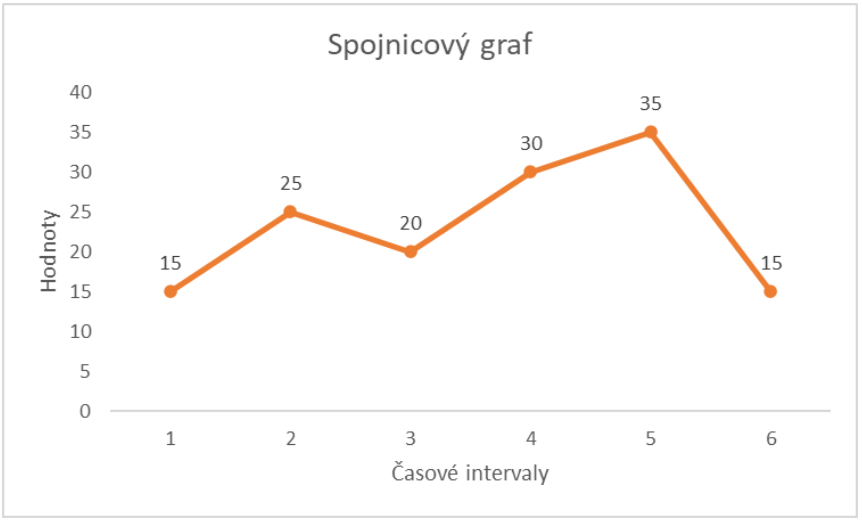
Stĺpcový graf je typ dátovej vizualizácie, ktorý používa zvislé alebo horizontálne stĺpce pre porovnanie kvantitatívnych hodnôt medzi rôznymi kategóriami v danom časovom období. Každý stĺpec reprezentuje jednotlivé kategórie, výška stĺpca odpovedá hodnote kategórie.




**Obrázok č. 10:** Stĺpcový graf (Vlastné spracovanie)

### 1.1.8.2 Spojnicový graf

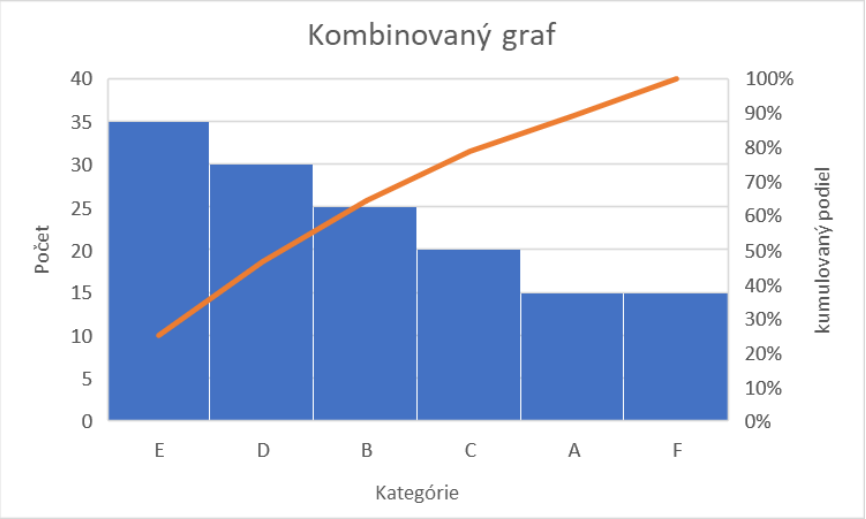
Spojnicový graf je typ vizualizácie dát, ktorý používa líniu alebo spojnicu k prezentácií série dátových bodov, zoradených podľa premennej, najčastejšie času. Spojnicové grafy sa aplikujú na zobrazenie trendov, zmien a vývoja v čase.




**Obrázok č. 11:** Spojnicový graf (Vlastné spracovanie)

### 1.1.8.3 Kombinovaný graf

Kombinovaný Paretov graf je vizualizácia, ktorá kombinuje stĺpcový a spojnicový graf v jednom. Stĺpcový graf vizualizuje jednotlivé hodnoty zoradené zostupných poradím. Spojnicový graf vizualizuje kumulatívny percentuálny súčet hodnôt. Používa sa na identifikáciu kľúčových prvkov v rámci dát.




**Obrázok č. 12:** Kombinovaný graf (Vlastné spracovanie)

#### 1.1.8.4 Tabuľka

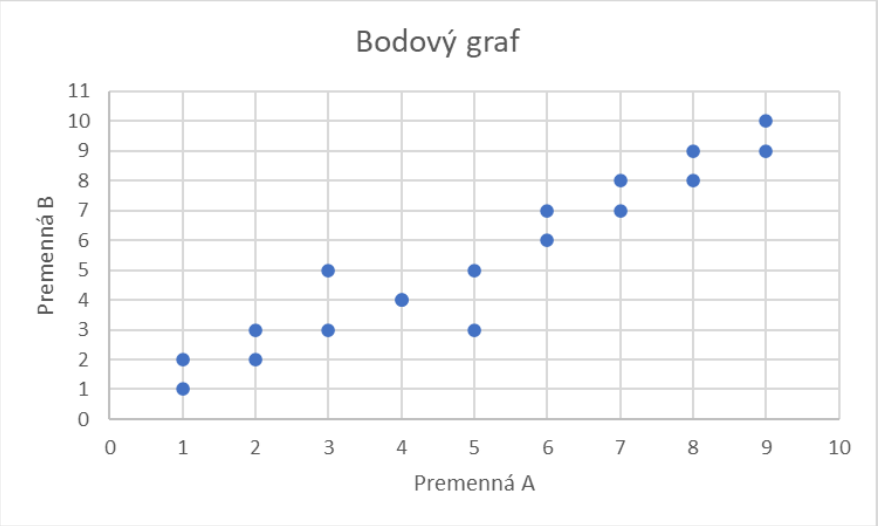
Tabuľky sú štruktúrovaný spôsob prezentácie informácií, v ktorých sa dáta organizujú do stĺpcov a riadkov. Každý riadok (alebo stĺpec) reprezentuje jednotlivé jednotky dát, stĺpce (alebo riadky) reprezentujú rôzne rozdelenia a kategórie informácií o týchto jednotkách. Spôsob využitia riadkov a stĺpcov je možné kombinovať podľa potreby. Tabuľky sa používajú na zobrazenie veľkého množstva dát na jednom mieste. Umožňujú tak rýchle a jednoduché porovnávanie dát.

#### 1.1.8.5 Text a obrázok

Texty a obrázky sú základnými elementy v reportingu, pomocou ktorých sa predávajú a interpretujú informácie a poskytuje sa kontext užívateľom. Texty zvyčajne obsahujú podrobné informácie a vysvetlenia. Obrázky sú realizované samotnými grafmi, tabuľkami, diagramami alebo fotografiami.

### 1.1.8.6 Bodový graf

Bodový graf je typ vizualizácie, ktorý používa body k zobrazeniu hodnôt dvoch premenných na horizontálnej a vertikálnej ose. Každý bod v grafe reprezentuje jeden dátový bod, ktorého poloha odpovedá hodnotách oboch premenných. Bodové grafy sa používajú na vizualizáciu vzťahov medzi premennými, napríklad korelácií, trendov, ale tiež odchýlok od očakávaných vzorcov.




**Obrázok č. 13:** Bodový graf (Vlastné spracovanie)

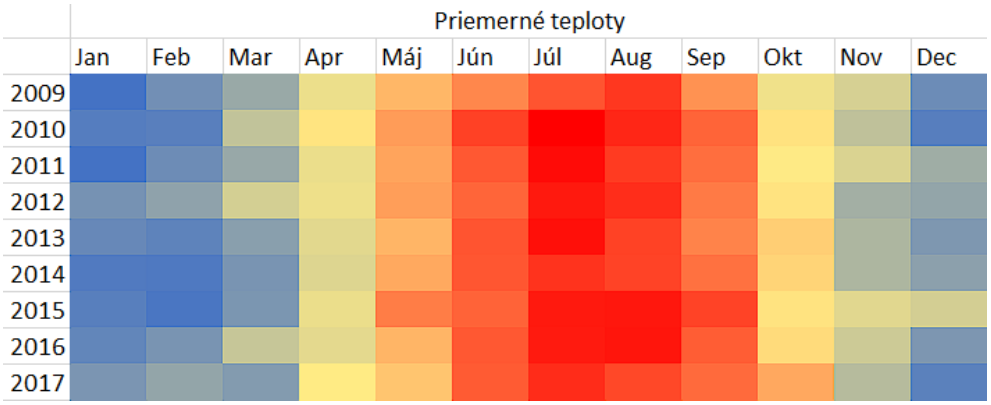
### 1.1.8.7 Výsekový graf

Výsekový graf, nazývaný aj ako koláčový graf, používa kruh rozdelené na segmenty (výseky), k zobrazeniu pomeru jednotlivých častí k celku. Každý výsek reprezentuje kategóriu dát, veľkosť výsek reprezentuje podiel kategórie na celkovom súčte. Výsekový graf sa používa na vizualizovanie relatívnych pomerov častí k celku.


**Obrázok č. 14:** Výsekový graf (Vlastné spracovanie)

### 1.1.8.8 Teplotná mapa

Teplotná mapa, alebo heatmapa, používa farebné stupnice k zobrazeniu hodnôt v matici alebo v tabuľke. Každá bunka v matici reprezentuje jeden dátový bod a farba bunka odpovedá hodnote bodu. Teplotné mapy sa používajú k vizualizácií komplexných dátových sád.

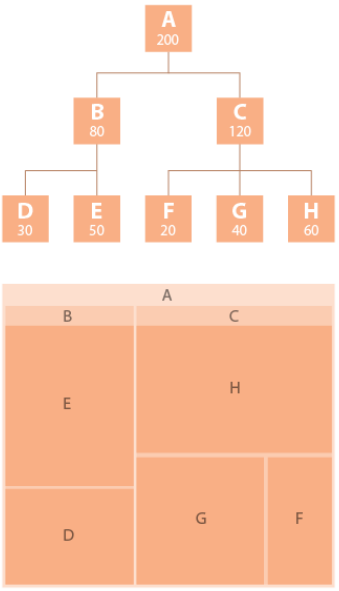



**Obrázok č. 15:** Teplotná mapa (Vlastné spracovanie)

### 1.1.8.9 Stromová mapa

Stromová mapa používa obdĺžniky rôznych veľkostí a farieb k prezentácií štruktúrovaných a hierarchických dát. Každý obdĺžnik vyjadruje určitú kategóriu dát,

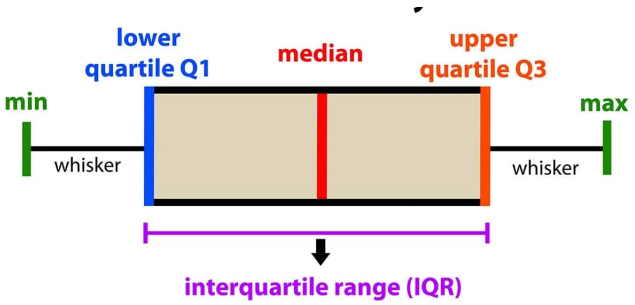
veľkosť a farba obdĺžnika odpovedá hodnote kategórie. Stromové mapy sa používajú na vizualizáciu hierarchických dát a porovnanie relatívnych veľkostí jednotlivých kategórií.




**Obrázok č. 16:** Teplotná mapa (Zdroj: [6])

### 1.1.8.10 Krabicový graf

Krabicový graf je typ grafu, ktorý sa používa k zobrazeniu rozdelenia a rozptylu hodnôt, odhalenie výkyvov, odľahlých hodnôt a symetrie v dátovej sade. Graf zobrazuje 5 kľúčových štatistických hodnôt, minimum, prvý kvartil (25.percentil), medián, tretí kvartil (75.percentil) a maximum.




**Obrázok č. 17:** Teplotná mapa (Zdroj: [7])

#### 1.1.8.11 KPI

Kľúčové ukazovatele výkonnosti (Key Performance Indicator - KPI) sú metriky, ktoré používajú organizácie k meraniu, sledovanie a vyhodnoteniu úspešnosti svojich podnikateľských aktivít, stratégií a cieľov. Sú základným nástrojom pre hodnotenie efektivity a účinnosti v rôznych oblastiach podnikania, ako je predaj, výroba, marketing atď. Kľúčové ukazovatele výkonnosti odpovedajú cieľom organizácie pre vyhodnotenie aktuálneho pokroku v porovnaní so svojimi cieľmi. Môžu byť kvantitatívne, napríklad počet predaných produktov, a kvalitatívne, napríklad spokojnosť zákazníkov. Zmysluplné KPI by mali byť jednoduché na pochopenie, relevantné pre organizáciu, merateľné a schopné jasne vysvetliť aktuálnu situáciu organizácie [8].

#### 1.1.8.12 Dashboard

Dashboard je súbor nástrojov pre vizualizáciu dát, ktoré agregujú a zobrazené kľúčové metriky a kľúčové ukazovatele výkonu na jednom mieste. Sú základným nástrojom pre riadenie informácií a znalosti v oblasti manažmentu organizácií, kde slúžia na sledovanie výkonnost, trendov a iných aspektov podnikových aktivít. Používajú sa na prezentáciu aktuálnych dát, zrovnávanie s historickými dátami a identifikáciu trendov v kľúčových oblastiach podnikania.

Dashboardy využívajú grafy, diagramy, tabuľky a iné formy vizualizácií v jednoduchej a zrozumiteľnej podobe pre zobrazenie a interpretovanie komplexných dátových sád. Pomocou nich je možné rýchle a efektívne rozhodovať na základe dát. Dashboardy môžu byť interaktívne, čím užívatelia môžu preskúmavať dáta, filtrovať podľa špecifických parametrov alebo prispôsobiť vizualizácie podľa vlastných potrieb.

Dashboardy sú kľúčové pre efektívne riadenie a monitorovanie podnikovej výkonnosti, ktoré umožňujú organizáciám rýchlo reagovať na zmeny a optimalizovať svoje aktivity na základe jasných a aktuálnych dát [9].

## 1.2 API

Táto časť práce čerpá informácie z [10]. API znamená aplikačné programovacie rozhranie (Application Programming Interface), je súbor pravidiel a špecifikácií, ktoré umožňujú komunikáciu a integrovanie medzi rôznymi softvérovými aplikáciami. Rozhranie API poskytuje abstrakciu, ktorá umožňuje oddeliť rôzne zložky softvérového systému. Takto môže jedna aplikácia používať funkcie alebo služby inej aplikácie bez toho, aby musela podrobne poznať jej vnútornú implementáciu a fungovanie. Táto abstrakcia je kľúčom k dosiahnutiu modularity, opätovnej použiteľnosti a rozšíriteľnosti v softvérovom inžinierstve.

Existujú rôzne typy rozhraní API: webové API rozhranie, operačné systémy, databázy a softvérové knižnice. Webové API, známe aj ako HTTP API alebo REST API, sú veľmi populárne v kontexte cloudových služieb, webových stránok a služieb, a mobilných aplikácií. Tieto API umožňujú komunikáciu a výmenu údajov medzi rôznymi aplikáciami cez internet pomocou štandardných protokolov, ako je HTTP. Medzi základné funkcie a príkazy, ktoré API používa, patria:

- GET - Používa sa na načítanie údajov zo služby. Je to operácia len na čítanie a nemení žiadne údaje na serveri.
- PUT - Používa sa na aktualizáciu existujúcich údajov na serveri. Vyžaduje, aby klient odoslal celý aktualizovaný záznam, nielen zmeny.
- POST - Používa sa na odosielanie údajov na server. Môže sa použiť napríklad na vytvorenie nového záznamu v databáze.
- DELETE - Slúži na odstránenie existujúcich údajov na serveri.

Počas HTTP komunikácie účastníci komunikácie posielajú medzi sebou takzvané stavové HTTP kódy. Najčastejšie používané kódy sú:

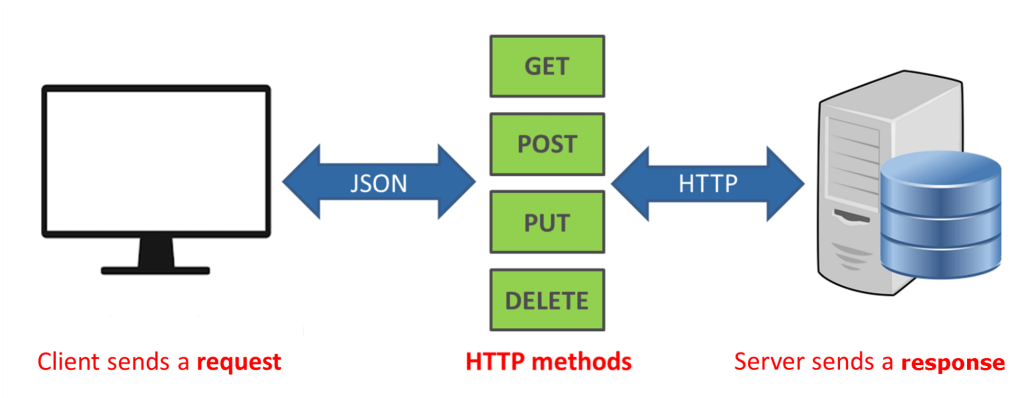
- 200 (OK) - Štandardná odpoveď pre úspešné požiadavky HTTP.
- 201 (Vytvorené): Požiadavka bola splnená a výsledkom bolo vytvorenie nového prostriedku.
- 204 (Žiadny obsah): Server úspešne spracoval požiadavku a nevracia žiadny obsah.

- 301 (Trvalo presunuté): Adresa URL požadovaného zdroja bola natrvalo zmenená. Nová adresa URL je uvedená v odpovedi.
- 302 (Nájdené): Server odpovedá týmto stavovým kódom, keď je požadovaný zdroj dočasne dostupný na inom URI.
- 400 (Zlá požiadavka): Server nerozumel požiadavke z dôvodu nesprávnej syntaxe.
- 401 (Neautorizované): Požiadavka vyžaduje overenie používateľa alebo, ak požiadavka obsahovala autorizačné poverenia, autorizácia bola pre tieto poverenia zamietnutá.
- 403 (Zakázané): Server požiadavke porozumel, ale odmietol ju autorizovať. Tento stavový kód sa bežne používa vtedy, keď server nechce prezradiť, prečo presne bola požiadavka odmietnutá, alebo keď nie je možné použiť inú odpoveď.
- 404 (Nebolo nájdené): Požadovaný zdroj sa na serveri nenašiel.
- 500 (Vnútorná chyba servera): Všeobecné chybové hlásenie, ktoré sa uvádza v prípade, že sa vyskytol neočakávaný stav a nie je vhodné žiadne konkrétnejšie hlásenie.
- 503 (Služba nedostupná): Server je momentálne nedostupný (preťažený alebo nefunkčný).

Rozhrania API môžu pracovať s mnohými rôznymi typmi údajov, ale najbežnejším formátom údajov používaným na odosielanie a prijímanie údajov prostredníctvom rozhraní API je formát JSON (JavaScript Object Notation), a to vďaka jeho ľahkej povahe a jednoduchosti, s akou sa dá vytvoriť a spracovať. Údaje JSON sú formátom, ktorý využíva ľudsky čitateľný text na prenos dátových objektov pozostávajúcich z dvojíc atribút-hodnota a polí. Príklad JSON objektu je nasledovný:

```
{
  "users": [
    {
      "id": 1,
      "name": "Name1 Surname1",
      "email": "name1.surname2@example.com"
    },
    {
      "id": 2,
      "name": "Name2 Surname2",
      "email": "name2.surname2@example.com"
    }
  ]
}
```

Ďalším bežným formátom údajov je XML (eXtensible Markup Language). Údaje XML sú svojou syntaxou podobné HTML a boli veľmi populárne pre API predtým, ako sa začal viac používať JSON. Okrem formátov JSON a XML existujú aj iné dátové formáty, ktoré možno používať s rozhraniami API, napríklad CSV (Comma Separated Values) a YAML (Yet Another Markup Language), hoci sú menej rozšírené.




**Obrázok č. 18:** Spôsob komunikácie cez API (Zdroj: [10])

## 1.3 Python

Python je zaradený medzi vysokoúrovňové programovacie jazyky. Je to dynamický, interpretovaný programovací jazyk s čistou a relatívne jednoduchou syntaxou, ktorá uľahčuje jeho učenie a použitie. Ponúka širokú škálu interných aj externých knižníc, umožňujúcich vytváranie sofistikovaných programov a algoritmov. Python je jazyk bez typov, čo znamená, že je možné priradiť akýkoľvek dátový typ k premennej a až na základe aktuálne priradenej hodnoty sa určuje, ako sa s danou premennou manipuluje. Dátové typy, ktoré Python používa, sú uvedené v nasledujúcej tabuľke [11].

**Tabuľka č. 1**: Dátové typy jazyka Python (Vlastné spracovanie)

| Typ | Názov | Príklad deklarácie |
|---|---|---|
| Prázdny typ | NULL | x=NONE |
| Celé číslo | integer | x=3 |
| Racionálne číslo | float | x=3.0 |
| Komplexné číslo | komplex | x=3.0 + 2j |
| Reťazec | string | x=”abc” |
| Zoznam | list | x=[1,2,3,”bc”] |
| N-tica | tuple | x=(3,6,2) |
| Set | set | x={“jablko”, “hruška”} |
| Slovník | dictionary | x={“štyri”: 4, “tri”: 3} |

**Zoznam** je organizovaná sekvencia prvkov. Predstavuje jeden z najčastejšie používaných dátových typov v Pythonu a je mimoriadne flexibilný.

**N-tica,** podobne ako zoznam, je usporiadaná sekvencia prvkov. Avšak, rozdiel medzi nimi je, že N-tica je nemenná. To znamená, že raz vytvorená N-tica sa nedá meniť alebo upravovať.
**Set** predstavuje neorganizovanú kolekciu jedinečných hodnôt alebo objektov.
**Slovník** je neusporiadaná sústava párov kľúč-hodnota. Je bežne používaný pri spracovaní veľkých dátových množstiev, pričom je optimalizovaný pre vyhľadávanie dát. Ak chcete získať hodnotu, musíte poznať príslušný kľúč.

### 1.3.1 SQLAlchemy

SQLAlchemy je open-source knižnica programovacieho jazyka Python. Poskytuje súbor nástrojov na interakciu s relačnými databázami. Knižnica kombinuje výkonné funkcie objektovo-relačného modelovania (ORM) a efektívne nízkoúrovňové databázové operácie (SQL dotazy).

Objektovo-relačné modelovanie (ORM) v rámci SQLAlchemy umožnuje manipulovať s dátami v databázach pomocou objektov a tried programovacieho jazyka Python namiesto ručného písania SQL dotazov. Znamená to, že umožňuje pracovať s databázami na vyššej úrovni abstrakcie, čím sa môže uľahčiť vývoj a údržba databáz. SQLAlchemy umožňuje manipuláciu s rôznymi databázovými systémami, ako napríklad PostgreSQL, MySQL, SQLite a podobne.

Na druhej strane, SQLAlchemy tiež poskytuje rozhranie a možnosť pre priame vytváranie a vykonávanie SQL dotazov, čím sú zachované výkonnosť a flexibilita SQL. Toto rozhranie poskytuje nástroje pre všetky typy databázových manipulácií [12].

### 1.3.2 Alembic

Alembic je databázový migračný nástroj pre SQLAlchemy. Poskytuje jednoduchý a intuitívny spôsob správy verzie databázy a vytvárať databázové migrácie v kontextu SQLAlchemy.

Alembic umožňuje vytvárať a spracovať “revízie”, ktoré zaznamenávajú zmeny v databázové štruktúre. Tieto revízie obsahujú operácie, akú se pridanie alebo odstránenie

tabuliek, stĺpcov a ďalších databázových prvkov. Alembic taktiež poskytuje nástroje pre automatické generovanie revízií na základe porovnania aktuálneho stavu databáze s definíciou modelu v prostredí SQLAlchemy.

Alembic podporuje rôzne typy databáz, rovnako ako SQLAlchemy. Vďaka integrácie s SQLAlchemy je Alembic užitočný u projektoch, kde je využívaný objektovo-relačné modelovanie (ORM) pre správu dátového modelu databáz [13].

## 1.4 Dátové modelovanie

Dátové modelovanie je proces v systémovom a softvérovom inžinierstvu, počas ktorého sa definujú požiadavky systému na štruktúru dát. Výsledkom dátového modelovania je dátový model. Dátové modely definujú štruktúru a formát dát, a vzťahy medzi jednotlivými dátovými objektami. Táto kapitola čerpá informácie z [2, 3, 14].

Každý reálny objekt zo sveta je možné reprezentovať dátovým objektom. Dátovými objektmi môžu byť napríklad autá, vodiči, školy, učitelia, študenti. Medzi týmito objektmi reálnom svete môžu byť vzťahy, ktoré je možné zachytiť aj v dátových modeloch. Každý reálny objekt má vlastnosti (atribúty), ktoré sú možné v rámci dátových modelov sledovať. Tieto vlastnosti objektov sú v dátovom modeli atribúty dátových objektov. Atribútami auta môžu byť napríklad rok výroky, farba, ŠPZ značka. Úlohou dátového modelovania je zachytiť tie reálne objekty, ktoré sú relevantné pre informačný systém. Tvorba a implementácia dátových modelov sa riadi tromi úrovňami abstrakcie:

- Konceptuálna úroveň: V rámci tejto úrovne sa definujú dátové objekty, ktoré sú relevantné pre informačný systém, vzťahy medzi objektmi a atribúty objektov, s ktorými bude systém pracovať.
- Technologická úroveň: V tejto úrovni sa určuje, ako budú dáta štruktúrované - v akom type databáze sa budú dáta ukladať, dátové typy jednotlivých atribút objektov.
- Implementačná úroveň: Táto úroveň sa zaoberá implementáciou dátového modelu a štruktúry dát v konkrétnom programovacom jazyku konkrétneho databázového systému.

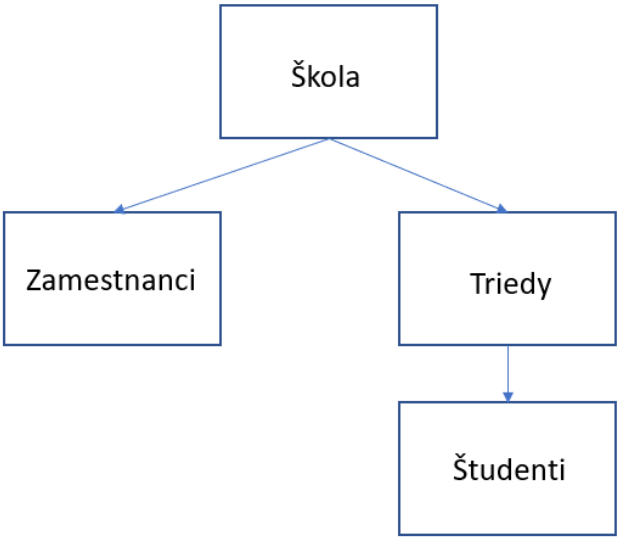
Podrobné informácie o dátovom modelovaní sa nachádzajú v [2, 3, 14].

## 1.5 Databázy

Databázy sú organizované súbory dát, ktoré sú špeciálne štruktúrované za účelom jednoduchého ukladania, spracovávania a vyhľadávania dát. Databázy sú súčasťou každého informačného systému. Databázy môžu obsahovať rôzne typy údajov - texty, čísla, obrázky, videá a ďalšie typy multimediálnych súborov. Tieto dáta môžu byť v databázach organizované a štruktúrované rôznymi spôsobmi, čo závisí od typu databázy. Podľa typu ukladania dát sú známe nasledujúce typy databáz [2, 14, 15]:

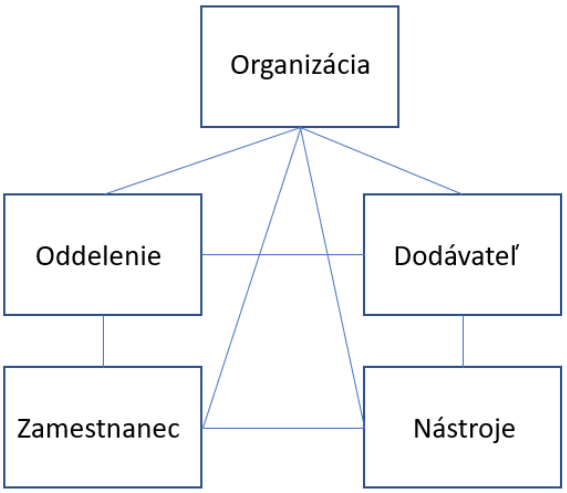
- hierarchické
- sieťové
- objektové
- relačné
- nerelačné

**Hierarchické databázy** sú založené na stromovej štruktúre. Každý záznam v databáze má jedného rodiča a nula alebo viac potomkov. Táto štruktúra odpovedá hierarchickej organizácií dát, pretože každá úroveň stromu je podmnožinou úrovne nad ňou.Výhodou hierarchických databáz je, že sú jednoduché na implementáciu a správu v operáciách, ktoré sledujú prirodzenú hierarchiu dát, napríklad databázy zamestnancov. Nevýhodou týchto databáz je malá flexibilita pri použitý zložitejších vzťahov medzi dátami.




**Obrázok č. 19:** Hierarchický dátový model (Vlastné spracovanie)

**V sieťových databázach** sú dáta organizované ako sieť záznamov, ktoré sú prepojené vzťahmi. Na rozdiel od hierarchických databáz záznamy môžu mať viacero rodičov. Táto vlastnosť umožňuje zachytiť zložitejšie vzťahy medzi objektmi z reálneho sveta. Nevýhodou týchto databáz je, že sú zložité na správu, pretože je potrebné dobre znať celkovú štruktúru databáze. Hierarchické a sieťové databázy sú najmenej používané databázy v moderných systémoch.

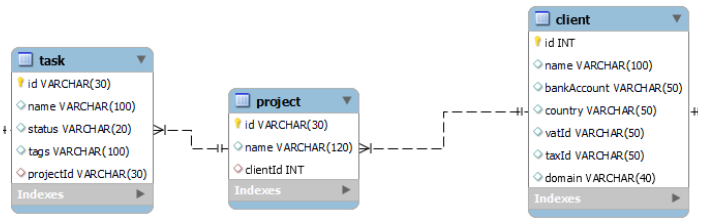



**Obrázok č. 20:** Sieťový dátový model (Vlastné spracovanie)

**V objektových databázach** sú dáta organizované ako súbor objektov. Každý objekt má svoju identitu, vlastnosti a metódy, ktoré definujú, ako sa s ním dá manipulovať. Objekty môžu byť zoskupené do trieb, ktoré môžu mať tiež svoje vlastnosti a metódy. Tieto databázy úzko súvisia v objektovo orientovaným programovaním. Poskytujú vysokú úroveň abstrakcie, pomocou ktorej je možné pracovať s komplexnými dátovými štruktúrami. Sú však zložité na správu, pretože neexistuje štandardizovaný dotazovací jazyk zrovnateľný s SQL u relačných databáz.

**Relačné databázy** sú najpoužívanejšie typy databáz. V rámci nich sú dáta organizované do tabuliek, ktoré sa volajú relácie. Každá tabuľka sa skladá z riadkov a stĺpcov, kde riadky predstavujú jednotlivé záznamy v tabuľke a stĺpce predstavujú atribúty záznamov. Tabuľky v relačných databázach môžu byť prepojené pomocou kľúčov, čo umožňuje vytvárať komplexné vzťahy medzi dátami. Jednotlivé atribúty v tabuľke môžu plniť funciu **primárneho, kandidátneho, alternatívneho alebo cudzieho kľúča**. V prípade kandidátneho kľúča sa jedná o atribúty, kroé jednoznačne určujú, o aký záznam v tabuľke sa jedná. Všetky atribúty, ktoré sú kandidátne kľúče, sa môžu stať primárny kľúčom. Tie, ktoré primárnym kľúčom nie sú, sa nazývajú alternatívne kľúče. Primárny kľúč je jeden alebo súbor atribútov, ktoré jednoznačne označuje záznam v tabuľke. Cudzí kľúč definuje vzťah medzi dvoma tabuľkami tým spôsobom, že hodnota v určenom stĺpci jednej tabuľky musí existovať v primárnom kľúči druhej tabuľky.

Relačné databázy sa tvoria a spravujú pomocou jazyka **SQL** (Structured Query Language). Vďaka štruktúrovanosti a konzistentnosti jazyka je možné záznamy v tabuľkách jednoducho filtrovať, triediť a uskutočňovať výpočty. Medzi konkrétne SQL relačné databázové systémy patria MySQL, Oracle, Microsoft SQL, MariaDB, PostgreSQL a mnoho ďalších.




**Obrázok č. 21:** Relačný dátový model (Vlastné spracovanie)

**Nerelačné databázy,** známe tiež ako NoSQL databázy, sa zásadne líšia od relačných databáz tým, že neukladajú dáta v pevne definovaných tabuľkách. Umožňuje to flexibilitu a kombináciu rôznych štruktúr dát bez nutnosti zásah do schématu databáze. Existuje niekoľko typov nerelačných databáz, ktoré sa odlišujú podľa toho, akým spôsobom pracujú s jednotlivými záznamami [19].

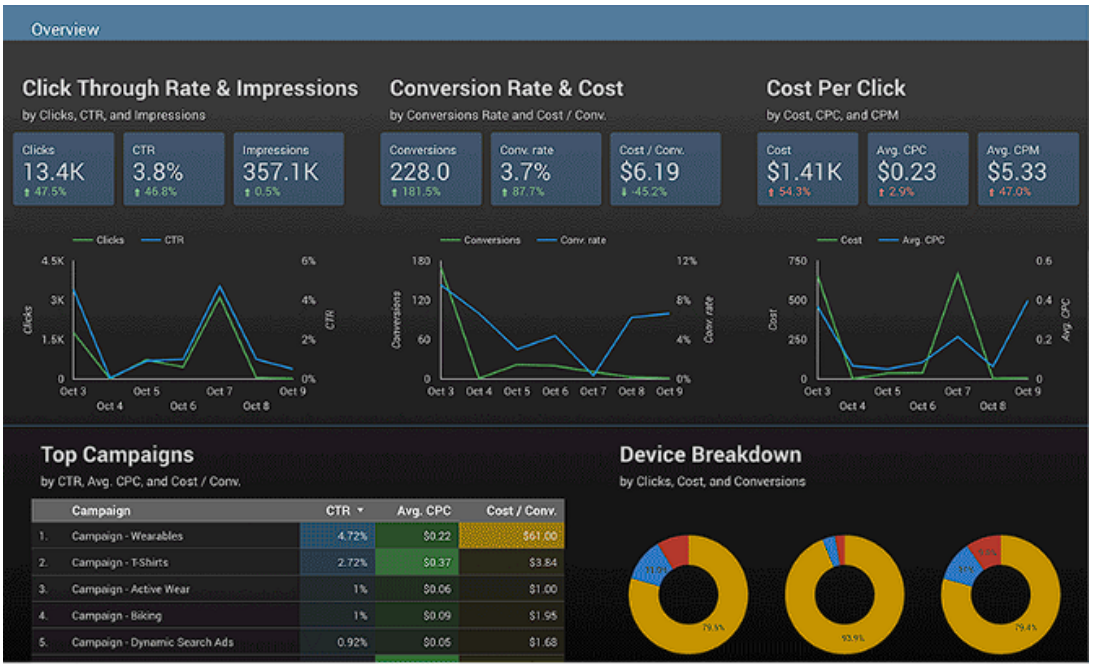
Najjednoduchším typov sú databázy na báze **kľúč-hodnota** (key-value), kde každému záznamu (hodnote) je priradený jedinečný identifikátor (kľúč), pomocou ktorého sa vyhľadáva. **Dokumentové** nerelačné databázy ukladajú údaje vo formáte dokumentu, najčastejšie vo formátoch JSON alebo XML. Výhodou je, že každý dokument je samostatný záznam, ktoré môže obsahovať rôzne typy a formát údajov s jedinečnou štruktúrou. Špeciálnymi typmi nerelačných databáz sú **stĺpcové,** ktoré ukladajú údaje podľa stĺpcov a nie riadkov, a **grafové**, ktoré sú založené na uzloch (konkrétne záznamy) a vzťahoch medzi nimi [19].

Najznámejšie systémy nerelačných databáz sú MongoDB, Apache Cassandra, Amazon Neptune.

## 1.6 Looker Studio

Looker Studio (predtým Google Data Studio) je interaktívny BI nástroj pre vizualizáciu dát a reporting. Nástroj umožňuje zbierať, analyzovať a vizualizovať dáta z rôznych zdrojov v reálnom čase.  Medzi podporované dátové zdroje patria Google Ads, Google Analytics, YouTube, BigQuery, MySQL, PostgreSQL a mnoho ďalších.

V rámci nástroja je možné vytvárať interaktívne reporty a dashboardy. Obsahuje mnoho typov grafov a diagramov. Reporty a dashboardy je možné prispôsobiť - nastaviť vzhľad, design, farby a rozloženie prvkov. Nástroj zahrňuje možnosť vytvárať vlastné vypočítané metriky a dimenzie podľa vlastných vzorcov a výrazov [16].




**Obrázok č. 22:** Dashboard v Looker Studio (Zdroj: [16])

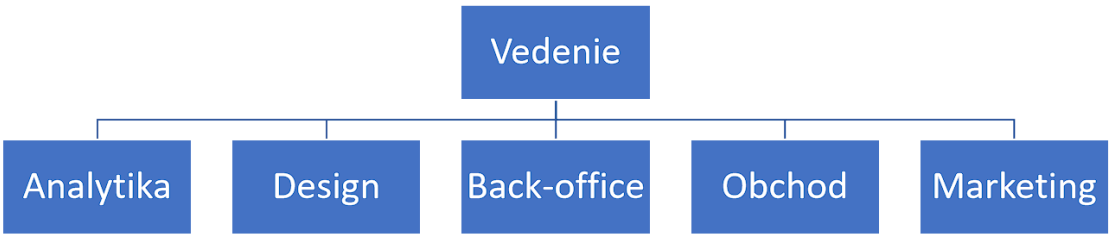
# 2 Analýza súčasného stavu

Analýza súčasného stavu spoločnosti sa skladá z troch hlavných častí. Prvá časť stručne predstavuje spoločnosť, v ktorej sa bude implementovať riešenie BI. Druhá časť analýzy sa venuje pochopeniu a zhrnutiu požiadaviek firmy na riešenie BI - technologických a informačných potrieb. Tretia časť sa zaoberá analýzou používaných systémov v spoločnosti a výberom zdrojových systémov, ktorých dáta sú relevantné pre integrovanie do dátového skladu.

## 2.1 Predstavenie spoločnosti

House of Řezáč s.r.o. je malá agentúra, ktorá pôsobí na trhu digitálnych služieb v oblasti UX/UI designu, marketingu, webovej a dátovej analytiky. Firma špecializuje svoje služby na webové stránky a webové aplikácie. Medzi hlavné produkty a služby, ktoré ponúka, patria: design a tvorba webových stránok a aplikácií, rozvoj strategického marketingu, webové a dátové analýzy, nastavenie merania chovania užívateľov stránok a aplikácií, optimalizácie konverzného pomeru webových stránok, školenia a workshopy.

Spoločnosť má v súčasnosti 18 zamestnancov. Hlavnú činnosť zamestnancov tvorí projektová práca.  V jednotlivých projektoch pracujú zvyčajne 2 až 3 zamestnanci analytického a designového oddelenia. Organizačná štruktúra spoločnosti je maticová. Nasledujúci obrázok zobrazuje organizačné štruktúru a jednotlivé oddelenia firmy.




**Obrázok č. 23:** Organizačná štruktúra  (Vlastné spracovanie)

## 2.2 Získanie požiadaviek zadávateľa

Pre získanie požiadaviek zadávateľa od riešenia Business Intelligence sa udiali rozhovory s vedením spoločnosti a kľúčovými budúcimi užívateľmi. Hlavným cieľom implementovaného riešenia je automatizovanie reportingu a vyriešenie problému pravidelnej neefektívnej práce zamestnancov pri zbere, vyhodnotení a interpretácií dát. Výsledky rozhovorov a požiadavky sú rozdelené do nasledujúcich oblastí: výroba, financie, obchod, HR. U každej z týchto oblastí sa nachádzajú metriky a KPI ukazovatele, ktoré chce spoločnosť sledovať.

### 2.2.1 Požiadavky na reporting výroby

Tieto požiadavky sa týkajú hlavných (projektových) činností firmy, ktoré sú vykonávané zamestnancami analytického a designového oddelenia. Požiadavky sú nasledujúce:

- počet odpracovaných hodín
- počet odpracovaných klientských (faktúrovateľných) hodín
- pomer klientskych a celkových hodín

### 2.2.2 Požiadavky na reporting financií

Požiadavky na reporting financií sa týkajú vydaných a prijatých faktúr spoločnosti. Požadované metriky na sledovanie sú:

- stav peňažných prostriedkov
- hodnota vystavených faktúr
- stav vystavených faktúr

### 2.2.3 Požiadavky na reporting obchodu

Tieto požiadavky sa týkajú obchodných aktivít spoločnosti. Patria sem obchodné požiadavky a ich posun v obchodnom lieviku. Požadované metriky na sledovanie sú:

- počet objednávok

- hodnota podaných ponúk
- hodnota odsúhlasených ponúk
- obchodné aktivity v čase

### 2.2.4 Požiadavky na reporting HR

Oddelenie HR zaujímajú aktivity náborového procesu. Chcú sledovať:

- počet nových životopisov
- stavy životopisov
- čas v jednotlivých stavoch životopisov

## 2.3 Mapovanie dátových zdrojov, používaných systémov a technológií

V spoločnosti bola prevedená analýza používaných informačných systémov, technológií, dát a vzťahov medzi nimi. V nasledujúcej časti podkapitoly sú predstavené a opísané používané systémy, v ktorých sú uložené hlavné dáta spoločnosti.

**ClickUp**

Hlavným systémom je informačný systém ClickUp. ClickUp je cloudový nástroj na spoluprácu a riadenie projektov. Systém bol zavedený v spoločnosti do práce koncom roku 2021. Spoločnosť predtým používala niekoľko oddelených systémov rôznych funkcionalít, ktorých dáta boli premigrované. To znamená, že v systéme sa nachádzajú historické dáta spoločnosti.

Hlavným využitím systému je **projektový manažment**. Spoločnosť tu riadi a archivuje všetky projekty, projektové úlohy, projektové rozpočty, plánovanie prác a vykazovanie odpracovaných časov na konkrétnych úlohách.

Systém sa tiež používa na evidovanie **obchodným partnerov a zákazníkov**. Obsahuje zoznam všetkých firiem, s ktorými firma niekedy spolupracovala a zoznam kontaktov firiem. Nachádzajú sa tu ak záznamy kontaktov jedincov, ktorí sa niekedy zúčastnili školení, prihlásili do odberu newsletteru alebo nejako inak sa angažovali v aktivitách firmy. Tieto kontakty a firmy sú previazané s konkrétnymi projektami.

V systému sa nachádza priestor na riadenie **obchodu**. Sú tu zaznamenané všetky historické obchodné aktivity. Obchodnými aktivitami sa rozumejú stavy obchodných objednávok, ich hodnota a úspešnosť obchodných aktivít. Úspešné obchodné projekty sú prepojené s konkrétnymi projektami.

Ďalším využitím systému je riadenie **náborové procesu**. Rovnako ako u obchodných procesov, sú tu zaznamenané profily uchádzaćov, prihlášky na pozície a aktivity v rámci náborového procesu.

V systému sú zaznamenané aj kópie jednotlivých vydaných a prijatých faktúr spoločnosti, napárované na konkrétne projekty alebo kontakty. Kópie faktúr vznikajú manuálnou prácou zamestnancov z jedného systému do druhého.

**SuperFaktura**

SuperFaktura je účtovnícky a fakturačný systém. Firma ju využíva na vystavovanie a evidovanie prijatých faktúr, evidenciu majetku a ostatných účtovníckych aktivít. Systém taktiež obsahuje zoznam firiem a kontaktov, ku ktorým sú pridelené jednotlivé faktúry. Tento systém je priamo integrovaný s **bankovým účtom** Fio banky, ktorá páruje prichádzajúce platby s faktúrami v systéme. Táto integrácia tiež slúžia na riadenie cash flow.

**Práca s dátami**

V spoločnosti so používa niekoľko foriem prezentácie a práce s dátami. Základné jednorázové reporty týkajúce sa financií a účtovníckych metrík sa vytvárajú v aplikácií Microsoft Excel, prípadne v Google Sheets. Zdrojmi dát sú dáta z predchádzajúcich opísaných systémov, ktoré sú ručne exportované a upravované. Tieto exporty sú obmedzené limitami exportov systémov.

**Looker Studio**

Na vizualizáciu a predávanie dát medzi oddeleniami, prípadné pokročilejšie analýzy sa používa vizualizačný nástroj Looker Studio (predtým Google Data Studio). Dôvod využívania tohto nástroja je ten, že je pravidelne využívaný v práci analytického oddelenia pre reportovanie dát z Google Analytics a Google Search Console. Výhody

tohto nástroja sú, že je to cloudové riešenie, je bezplatné, obsahuje množstvo dátových konektorov a je možné jednoducho zdieľať a pristupovať k reportom z dôvodu webového rozhrania.

**Interný NAS server**

V spoločnosti sa nachádza interný NAS server, ktorý slúži primárne ako úložisko väčších súborov ale taktiež na ňom bežia kontejnerizované aplikácie a databázy pomocou softvéru Docker pre interné testovanie a vývoj v analytickom oddelení.

Vzhľadom na to, že v spoločnosti je nástroj Looker Studio využívaný na tvorbu reportov, zamestnanci a vedenie firmy majú skúsenosti s jej použitím, spoločnosť nepripúšťa výber iných konkurenčných nástrojov ako komponent Business Intelligence pre reporting.

### 2.3.1 Možnosti exportu dát zo systému ClickUp

Pre potreby integrovania dát zo zdrojového systému ClickUp je potrebné poznať možnosti exportu týchto dát. Hlavnou vlastnosťou ETL procesov je automatizovaný prenos dát, ktorý požaduje minimálny manuálny zásah počas fungovania. Z toho dôvodu v tejto analýze nebudú rozobrané možnosti manuálneho exportu dát zo systému ClickUp v podobe CSV alebo XLSX [17].

V čase analýzy zdrojového systému jediný možný nemanuálny export dát je dostupné API rozhranie systému. Vybrané vlastnosti a technické obmedzenia API rozhrania systému ClickUp relevantné pre návrh a implementáciu Business Intelligence sú:

- **Dostupnosť:** Rozhranie API systému ClickUP je dostupné každému užívateľovi systému. Pomocou API je možné exportovať všetky dáta, s ktorými sa priamo alebo nepriamo pracuje v užívateľskom rozhraní (projekty, vykázané časy, úlohy).
- **Autentizácia:** Pre používanie API je potrebné dotazy autentizovať. ClickUP API poskytuje dva spôsoby autentizácie requestov: pomocou užívateľského tokenu a OAuth2 flow. Všetky potrebné konfiguračné tokeny je potrebné vygenerovať v rámci užívateľského účtu.

- **Formát dát:** API pracuje s JSON formátom dát. Jednotlivé prvky v rámci systému (projekty, úlohy, kontakty) sú uložené v štandardizovaných JSON objektoch, ktoré majú spoločné atribúty (jednoznačný identifikátor, názov, umiestnenie,...) a vlastné atribúty špecifické pre prvok, ktoré sú definované užívateľmi systému.
- **Relačnosť dát:** JSON objekty obsahujú atribúty o nadradených prípadne previazaných prvkoch v rámci systému.
- **História zmeny dát:** JSON objekty obsahujú atribúty s dátumami o vytvorení a posledných zmenách prvkov systému. API poskytuje špeciálnu funkcionalitu pre zistenie histórie zmien stavov úloh s časmi, ako dlho boli v jednotlivých stavoch.
- **Limity:** Existujú limity na počet API dotazov za minútu podľa predplatené plánu systému. Pre aktuálne nastavenie predplatného firmy je platný limit 100 API dotazov za minútu.

### 2.3.2 Možnosti exportu dát zo systému SuperFaktura

Pre potreby integrovania dát zo zdrojového systému SuperFaktura je potrebné poznať možnosti exportu týchto dát. V čase analýzy zdrojového systému existuje niekoľko priamych natívnych integrácií systému SuperFaktura a ostatných skladových, eshopových, účtovníckych a bankových aplikácií. Medzi týmito integrácie však nie sú žiadne relačné databázy. Okrem týchto integrácií má systém API rozhranie [18]. Vybrané vlastnosti a technické obmedzenia API rozhrania systému SuperFaktura relevantné pre návrh a implementáciu Business Intelligence sú:

- **Dostupnosť:** Rozhranie API systému je dostupné každému užívateľovi systému. Pomocou API je možné exportovať všetky dáta, s ktorými sa pracuje v užívateľskom rozhraní (faktúry, klienti).
- **Autentizácia:** Pre používanie API je potrebné autentizovať dotazy pomocou personálnych tokenov, dostupné v užívateľských účtoch systému.
- **Formát dát:** API pracuje s JSON formátom dát.
- **Relačnosť dát:** JSON objekty faktúr obsahujú hodnoty s identifikátorom klientov faktúr.

- **História zmeny dát:**API poskytuje aktuálny stav dát.
- **Limity:** Existujú pevné limity na poćet API dotazov. Denný limit je 1000 dotazov, mesačný limit je 30000.

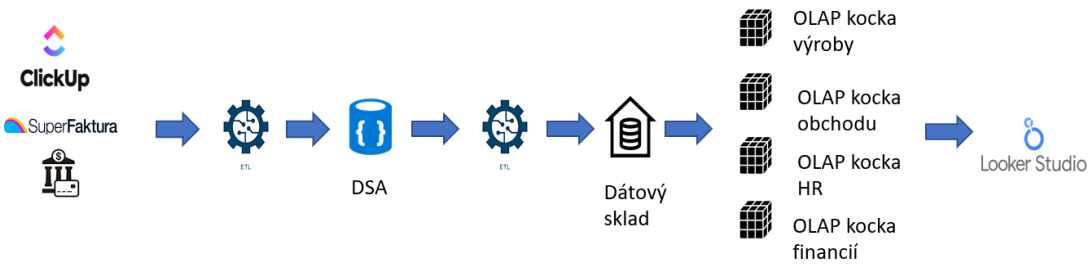
# 3 Vlastný návrh riešenia

Táto časť práce obsahuje komplexný návrh riešenia Business Intelligence pre riešenú firmu. Na základe informačných potrieb firmy a analýze zdrojových systémov sú navrhnuté jednotlivé komponenty architektúry BI, opísané konkrétne technologické využitia a implementácia komponentov použitím týchto technológií. Výber technologických riešení rešpektuje aktuálne využívané technológie v spoločnosti.

## 3.1 Návrh architektúry Business Intelligence

Po vyhodnotení Analýzy súčasného stavu a potrieb firmy navrhovaná architektúra Business Intelligence má nasledujúce komponenty:

- Zdrojové dáta: ClickUp, SuperFaktura a bankový účet.
- Vybrané dáta z týchto systémov budú pomocou ETL procesov extrahované a načítané do DSA.
- DSA - dočasné úložisko dát.
- Dáta z DSA budú pomocou ETL procesov extrahované, transformované a načítané do dátového skladu.
- Centrálny celopodnikový dátový sklad.
- OLAP kocky s príslušnými dátmi podľa potrieb na reporting.
- Následné načítanie a vizualizácia týchto dát bude prebiehať v nástroji Looker Studio.




**Obrázok č. 24:** Návrh architektúry BI (Vlastné spracovanie)

### Zdrojové dáta

Systémy, z ktorých budú dáta extrahované do dátového skladu sú ClickUP, SuperFaktura a bankový účet. Z analýzy informačných systémov vyplýva, že všetky tri systémy podporujú export dát pomocou API, to znamená, že je možné sa dostať jednoducho k dátam. Zo systému ClickUp bude exportovaných väčšina dát: vykazované časy, projekty a úkoly, dáta týkajúce sa obchodného a náborového procesu, zoznam zamestnancov a klientov. Z fakturačného systému SuperFaktura budú exportované dáta týkajúce sa faktúr a klientov. Z bankového účtu budú exportované stavy účtu ku konci dňa.

### ETL procesy

ETL procesy budú technicky riešené vlastnými Python skriptami pomocou knižníc Alembic, ktorý slúži na verzovanie databáz, SQLAlchemy, ktorý slúži na prácu s relačnými databázami a PyMongo, ktorý slúži na prácu s nerelačnými databázami. Tieto skripty budú orchestrované a automatizovane spúšťané pomocou nástroja Jenkins, ktorý slúži na automatizované spúšťanie procesov. Softvér Jenkins je hostovaný a dlhodobo využívaný na internom NAS serveru spoločnosti.
Dôvody výberu implementácie ETL procesov vlastným riešením pomocou Python skriptov sú nasledujúce:

- Väčšina známych nástrojov pre ETL procesy a dátové pumpy nepodporujú a nedisponujú konektormi na systémy ClickUp a SuperFaktura. Nástroje síce ponúkajú možnosti generických konektorov a vývoj vlastných konektorov, ktoré sú v záveru vlastné skripty.
- Podnik používa v zdrojových systémoch (ClickUP, SuperFaktura) vlastné polia (custom fields) na ukladanie dát, ktoré sú v rámci API rozhraní uložené v komplikovaných a vnorených štruktúrach. Práca s tými štruktúrami vyžaduje špecifické riešenia.
- V rámci ETL procesov je potrebné záznamy klientov deduplikovať a spojovať, na základe špecifickej logiky, ktorú je nutné vlastnoručne implementovať.

**DSA - dočasné úložisko dát**

Po extrahovaní dát zo zdrojových systémov budú dáta dočasne uložené v dočasnom úložisku dát - DSA. Toto úložisko bude mať podobu nerelačnej databáze MongoDB, ktorý je schopný ukladať surové JSON dokumenty. Databáza MongoDB bude hostovaná na internom NAS serveru. Ak by v budúcnu bolo potrebné dáta zo zdrojových systémov zálohovať, časť tejto nerelačnej databáze môže slúžiť ako záloha dát.

**Dátový sklad**

Pre rozsah a objem dát spoločnosti a potreby pre reportovanie informácií naprieč relatívne malé oddelenia bude vybudovaný centrálny celopodnikový dátový sklad. Dátový sklad bude mať podobu relačnej databáze, technologicky riešený PostgreSQL databázou. Táto databáza bude hostovaná na internom NAS serveri.

**OLAP kocky a reporting**

Vizualizovanie dát a reporty budú vytvorené v nástroji Looker Studio. Pre prácu a integrovanie dát do nástroja z dátového skladu budú vytvorené OLAP kocky v podobe multidimenzionálnych tabuliek - databázových pohľadov (views). Tieto tabuľky budú integrované do vizualizačného nástroja pomocou zabudovaného konektoru pre PostgreSQL databázy.

## 3.2 Dátový model a dimenzionálne modelovanie

Navrhovaný dátový model a použité dimenzie vychádzajú z analýzy požiadaviek zadávateľa, z analýzy používaných informačných systémov a dát, ktoré sa v nich nachádzajú a s ktorými spoločnosť pracuje.
Navrhovaný dátový model obsahuje dimenzionálne tabuľky, ktoré sú zobrazené v nasledujúcej tabuľke.

**Tabuľka č. 2:** Zoznam navrhovaných dimenzionálnych tabuliek (Vlastné spracovanie)

| Tabuľka | Názov dimenzie | Popis |
|---|---|---|
| employee | zamestnanec | Pohľad na dáta podľa jednotlivých zamestnancov |
| task | úloha | Pohľad na dáta podľa jednotlivých projektových úloh |
| project | projekt | Dimenzia poskytuje pohľad podľa projektov |
| client | klient | Dimenzia poskytuje pohľad podľa klientov |
| candidate | kandidát | Pohľad na dáta podľa jednotlivých kandidátov |
| status_hr | HR stav | Dimenzia slúži ako číselník stavov životopisov v náborovom procese |
| sales_lead | objednávka | Pohľad na dáta podľa jednotlivých objednávok |
| status_sales | obchodný stav | Dimenzia slúži ako číselník stavov objednávok |

Návrhy atribútov predchádzajúcich dimenzionálnych tabuliek sú opísané v nasledujúcej časti.

**Tabuľka employee**

Tabuľka obsahuje zoznam všetkých historických a stávajúcich zamestnancov firmy. Jeden záznam v tabuľke je jeden zamestnanec. Zdrojové dáta tejto tabuľky pochádzajú zo systému ClickUp, kde sú zamestnanci evidovaní. Atribúty sú:

- id - VARCHAR(20) - primárny kľúč
- name - VARCHAR(100) - meno zamestnanca

- primaryJob -VARCHAR(30) - pozícia zamestnanca
- email - VARCHAR(50) - email zamestnanca

**Tabuľka task**

Tabuľka obsahuje zoznam projektových úloh, ktoré sa nachádzajú v systéme ClickUp. Jeden záznam projektu je jedna unikátna úloha na projektu. Obsahuje nasledujúce atribúty:

- id - VARCHAR(30) - primárny kľúč
- name - VARCHAR(100) - názov úlohy
- status - VARCHAR(20) - stav úlohy
- tags - VARCHAR(100) - spojené štítky projektových úloh v nasledujúcom formáte: *ŠTÍTOK1,ŠTÍTOK2,ŠTÍTOK3*
- projectId - VARCHAR(30) - cudzí kľúč tabuľky project

**Tabuľka project**

Táto tabuľka obsahuje zoznam všetkých projektov vo firme. Zdrojové dáta sa nachádzajú v systéme ClickUp. Atribúty sú:

- id - VARCHAR(30) - primárny kľúč
- name - VARCHAR(120) - názov projektu
- clientId - INT - cudzí kľúč tabuľky client

**Tabuľka client**

Tabuľka client obsahuje zoznam všetkých klientov spoločnosti - firmy a jednotlivci. Zdrojové dáta tejto tabuľky sú systémy ClickUp a SuperFaktura. Táto tabuľka bude obsahovať rádovo nižšie tisícky záznamov. Atribúty sú:

- id - INT - primárny kľúč
- name - VARCHAR(100) - názov klienta
- bankAccount - VARCHAR(50) - číslo účtu klienta
- country - VARCHAR(50) - krajina klienta
- vatId - VARCHAR(50) - DIČ klienta
- taxId - VARCHAR(50) - IČO klienta
- domain - VARCHAR(50) - hlavná webová doména klienta

**Tabuľka candidate**

Tabuľka obsahuje zoznam kanditátov a pozíc, na ktoré sa hlásia. Zdrojové dáta tejto tabuľky sa nachádzajú v systéme ClickUp. Predpokladané množstvo je nižšie desiatky záznamov v tabuľke. Obsahuje atribúty:

- id - VARCHAR (40) - primárny kľúč
- name - VARCHAR(100) - meno kandidáta
- accepted - BOOLEAN - označuje, či kandidát bol prijatý
- position - VARCHAR(30) - názov pozície, na ktorú sa kandidát hlásil

**Tabuľka status_hr**

Táto tabuľka slúži ako číselník - obsahuje výčet statov, ktoré používa HR oddelenie v náborovom procese. Zdrojové dáta sú v systéme ClickUP. Obsahuje 2 atribúty:

- id - INT - primárny kľúč
- name - VARCHAR(100) - názov stavu

**Tabuľka sales_lead**

Táto tabuľka obsahuje zoznam všetkých objednávok, ktoré sa evidujú v obchodnom procese. Tieto dáta sa nachádzajú v systéme ClickUp. Atribúty sú nasledujúce:

- id - VARCHAR(40) - primárny kľúč
- name - VARCHAR(100) - názov objednávky
- won - BOOLEAN - označuje, či objednávka dopadla úspešne
- hourRate - FLOAT - hodinová sadzba projektu
- offerMAX - FLOAT - maximálna hodnota ponuky
- offerMin - FLOAT - minimálna hodnota ponuky
- offerFinal - FLOAT - konečná dohodnutá hodnota ponuky
- designerAmount - INT - počet zamestnancov designérov na projekte
- analystAmount - INT - počet zamestnancov analytikov na projekte

**Tabuľka status_sales**

Táto tabuľka, obdobne ako status_hr, obsahuje zoznam stavov objednávok v obchodnom procese. Obsahuje atribúty:

- id - INT - primárny kľúč

- name - VARCHAR(100) - názov stavu

Dátový model ďalej obsahuje 5 tabuľky faktov, opísané v nasledujúcej tabuľke.

**Tabuľka č. 3:** Zoznam navrhovaných dimenzionálnych tabuliek (Vlastné spracovanie)

| Tabuľka | Názov faktu | Popis |
|---|---|---|
| timelog | vykázaný čas | Tabuľka obsahuje jednotlivé záznamy vykázaných časov |
| transaction | faktúry | Tabuľky obsahuje vydané a pridané faktúry firmy |
| candidate_status | stavy životopisov | Tabuľka obsahuje záznamy stavov jednotlivých životopisov |
| sales_lead_status | stavy objednávok | Tabuľka obsahuje záznamy statov jednotlivých objednávok |
| account_balance | stav účtu | Tabuľka obsahuje stavy bankového účtu |

Tieto tabuľky faktov majú nasledujúce atribúty:

**Tabuľka timelog**

Táto tabuľka obsahuje jednotlivé záznamy odpracovaných časov, ktoré boli zadané zamestnancami do systému ClickUp. Časové záznamy sú vykazované na konkrétne úlohy v konkrétnych projektoch príslušnými zamestnancami. Tabuľka obsahuje tieto atribúty:

- id - VARCHAR(30) - primárny kľúč
- employeeId - VARCHAR(20) - cudzí kľúč tabuľky employee
- taskId - VARCHAR(30) - cudzí kľúč tabuľky task

- tags - VARCHAR(20) - obsahuje štítok vykázaného času
- startDate - DATETIME - čas začiatku vykázaného času
- endDate - DATETIME - čas konca vykázaného času
- updateDate - DATETIME - dátum poslednej úpravy záznamu
- durationMs - trvanie odpracovaného času
- isBillable - BOOLEAN - označenie, či je čas faktúrovateľný
- removed - BOOLEAN - označenie, či záznam bol odstránený zo systému ClickUp

**Tabuľka transaction**

Tabuľka transaction obsahuje všetky vydané a prijaté faktúry spoločnosti. Zdrojové dáta tabuľky sa nachádzajú v systéme SuperFaktura a v systéme ClickUp. Obsahuje atribúty:

- id - VARCHAR(20)
- clientId - INT - cudzí kľúč tabuľky client
- currency - VARCHAR(4) - mena faktúry
- amount - FLOAT - hodnota faktúry bez DPH
- amount_with_VAT - FLOAT - hodnota faktúry s DPH
- vat - FLOAT - hodnota DPH
- type - VARCHAR(64) - typ transakcie: prijatá alebo vydaná faktúra
- name (VARCHAR 200) - názov faktúry
- transaction_number - VARCHAR(30) - číslo faktúry
- date - DATE - dátum zdaniteľného plnenia faktúry
- status - VARCHAR(20) - stav zaplatenia faktúry
- txService - VARCHAR(100) - typ služby
- txCategory - VARCHAR(100) - kategória služby
- txSubcategory - VARCHAR(100) - podkategória služby

**Tabuľka candidate_status**

Tabuľka obsahuje historické záznamy zmien stavov životopisov kandidátov. Zdrojová dáta na nachádzajú v systéme ClickUp. Atribúty tejto tabuľky sú:

- id - INT - primárny kľúč
- statusId - INT - cudzí kľúč tabuľky status_hr

- candidateId - VARCHAR(40) - cudzí kľúč tabuľky candidate
- duration - INT - časová dĺžka životopisu v stave v milisekundách
- date_since - DATETIME - dátum a čas posunu životopisu do príslušného stavu
- is_last_status - BOOLEAN - označuje, či kandidát ukončil náborový proces v konkrétnom stave.

**Tabuľka sales_lead_status**

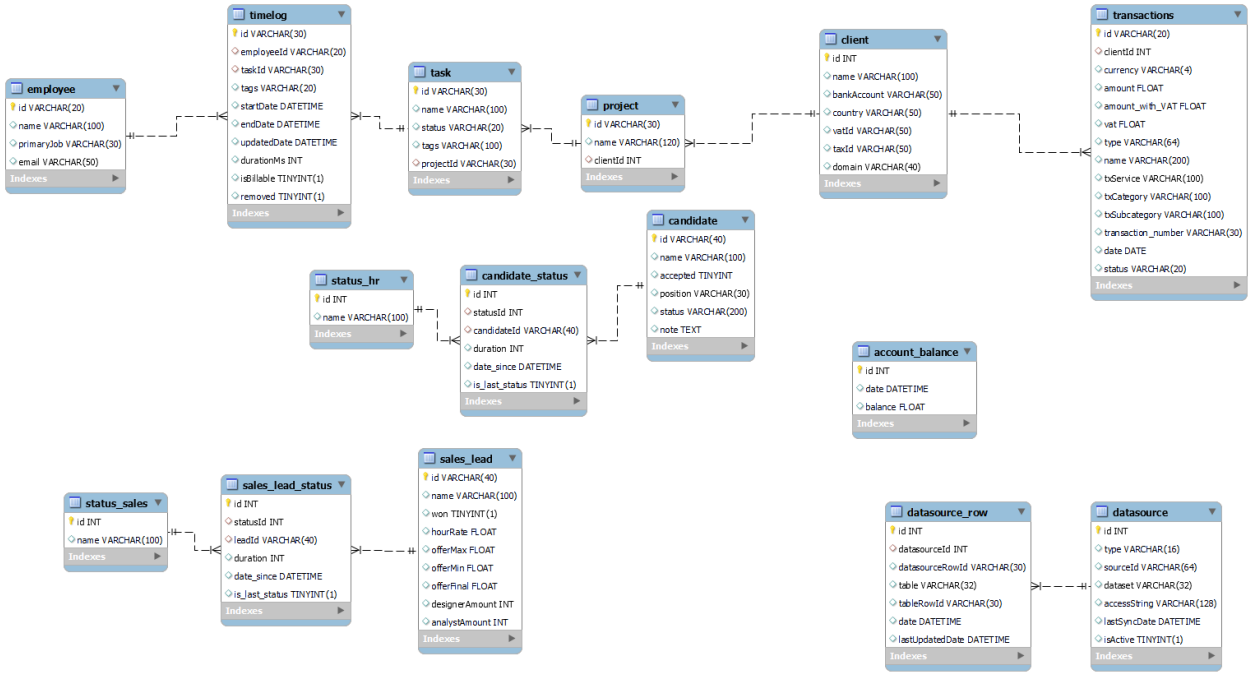
Tabuľka obsahuje historické záznamy zmien stavov objednávok v obchodnom procese. Zdrojová dáta na nachádzajú v systéme ClickUp. Atribúty tejto tabuľky sú:

- id - INT - primárny kľúč
- statusId - INT - cudzí kľúč tabuľky status_sales
- leadId - VARCHAR(40) - cudzí kľúč tabuľky sales_lead
- duration - INT - časová dĺžka objednávky v stave obchodného procesu v milisekundách
- date_since - DATETIME - dátum a čas posunu objednávky do príslušného stavu
- is_last_status - BOOLEAN - označuje, či objednávka sa ukončila v príslušnom stave

**Tabuľka account_balance**

Táto tabuľka obsahuje stav bankového účtu ku konci každého dňa. Zdrojové dáta je API rozhranie konkrétnej banky. Obsahuje atribúty:

- id - INT - primárny kľúč
- date - DATETIME - dátum a čas
- balance - FLOAT - stav účtu




**Obrázok č. 25:** Dátový model navrhovaného dátového skladu (Vlastné spracovanie)

### 3.2.1 Pomocné tabuľky

V rámci dátového modelu sú vytvorené dve navzájom prepojené pomocné tabuľky - datasource a datasource_row. Tieto tabuľky slúžia ako archivačné tabuľky záznamov po spojení a deduplikovaní záznamov týkajúcich sa rovnakého objektu (napríklad klienta) z viacerých dátových zdrojov.

Tabuľka datasource obsahuje záznam všetkých typov dátových zdrojov, z ktorých boli záznamy deduplikované a spojené. Tabuľka datasource_row označuje konkrétny záznam s dátového zdroja, ktorý vstúpil do deduplikácie alebo spojenia s iným záznamom.

Tabuľka datasource_row nesie informácie o pôvodných identifikátoroch databázového záznamu a informáciu o stávajúcom zázname, ktorý vznikol deduplikovaním prípadne spojením viacerých záznamov.

## 3.3 Návrh a implementace ETL

Táto kapitola nasleduje návrh architektúry Business Intelligence a návrh dátového modelu dátového skladu. Opisuje návrh a vytvorenie ETL procesov pomocou Python skriptov. Kapitola obsahuje prerekvizity k implementácií ETL procesov.

### 3.3.1 Predpoklady implementácie ETL

K implementácií ETL procesov je potrebné previesť určité operácie, ktoré sú nevyhnutné pre prostredie implementácie. Nasledujúce podkapitoly obsahujú opis potrebných operácií

#### 3.3.1.1 Vytvorenie dátového modelu

Pre vytvorenie dátového modelu relačnej databáze bude použitá Python knižnica SQLAlchemy a pre riadenie verzií dátového modelu dátového skladu nástroj Alembic.
V rámci Python skriptov je potrebné vytvoriť Python triedy použitím knižnice SQLAlchemy pre každú tabuľku v relačnej databáze (dátovom sklade) zvlášť. Nasledujúce obrázky ukazujú spôsob vytvorenia tabuľky faktov transactions a dimenzionálnej tabuľky. Pre každú tabuľku v dátovom sklade existuje vlastná trieda.

```
class Transaction(Base):
    __tablename__ = 'transactions'
    __table_args__ = (
    UniqueConstraint('id','type', name="unique_transaction")
    )

    id = Column(String(20), primary_key=True)
    clientId = Column(Integer, ForeignKey('client.id'))
    currency = Column(String(4))
    amount = Column(Float)
    amount_with_VAT = Column(Float)
    vat = Column(Float)
    type = Column(String(64))
    name = Column(String(200))
    transaction_number = Column(String(30))
    date = Column(Date)
    status = Column(String(20))
    txService = Column(String(100))
    txCategory = Column(String(100))
    txSubcategory = Column(String(100))
```

**Obrázok č. 26:** Deklarácia tabuľky faktov v SQLAlchemy (Vlastné spracovanie)

```python
class Project(Base):
    __tablename__ = 'project'

    id = Column(String(30), primary_key=True)

    clientId = Column(Integer, ForeignKey('client.id'))

    name = Column(String(120))
```

**Obrázok č. 27:** Deklarácia dimenzionálnej tabuľky v SQLAlchemy (Vlastné spracovanie)

#### 3.3.1.2 Migrácia dátového modelu

Po vytvorení tried pre každú databázovú tabuľku a konfigurácií nutných súborov je možné previesť prvotnú migráciu dátového modelu do dátového skladu. Pre vytvorenie migračného súboru je potrebné zavolať nasledujúci kód v konzoli:

*alembic revision -m "Create initial migration"*

Tento príkaz vytvorí migračný súbor typu .py, ktorý zahrňuje všetky informácie o samotnej migrácií - vytvorené, zmenené, odstránené tabuľky a stĺpce, unikátny identifikátor samotnej migrácie a unikátny identifikátor predošlej verzie, z ktorej migrácia vychádza. Pomocou upgrade a downgrade príkazov je možné verzie dátového skladu riadiť podľa verzií. Pre prvotnú migráciu dátového modelu do dátového skladu je potrebné zavolať nasledujúci kód:

*alembic upgrade head*

Týmto kódom sa v databáze vytvorí oddelená tabuľka s názvom alembic_version. Táto tabuľka má jediný stĺpec s názvom version_num a jediný záznam - aktuálny jedinečný identifikátor verzie dátového modelu.

#### 3.3.1.3 Vytvorenie spojenia so zdrojovými systémami

Pre úspešnú implementáciu ETL procesov je potrebné zaistiť autorizované pripojenie k dátam pomocou API rozhraní systémov. V jednotlivých systémov je potrebné

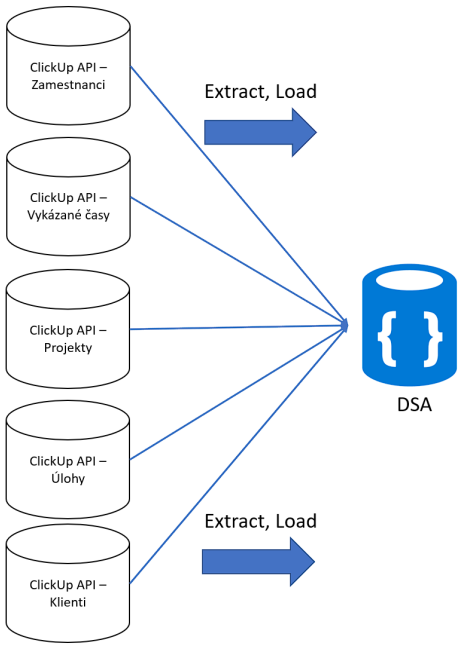
vygenerovať tzv. API tokeny, prípadne zaregistrovať profily, ktoré budú mať prístup k API rozhraniam.

### 3.3.2 Návrh ETL procesov

Táto časť podkapitoly opisuje logický návrh ETL procesov, ako jednu z komponent Business Intelligence.

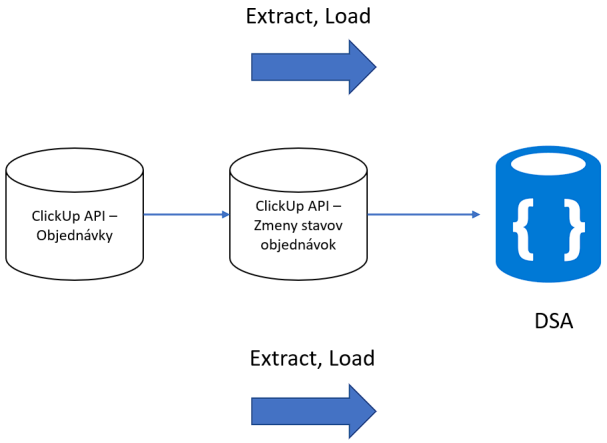
#### 3.3.2.1 Naplnenie DSA

Pre extrahovanie a naplnenie dát do dočasného úložiska dát (DSA) bude použitých 5 separátnych ETL procesov. Tieto procesy môžu byť spúšťané časovo nezávisle od seba. Prvý ETL proces má za úlohu extrahovať a načítať dáta o zamestnancoch, vykázaných časoch, projektoch, úlohách a klientoch z API rozhrania systému ClickUp do DSA. Dáta, ktoré poskytuje API sú vo forme JSON, takže tieto surové dáta sú možné uložiť do nerelačnej databáze ako separátne dokumenty. Všetky štyri aktivity v rámci ETL procesu nie sú navzájom závislé.

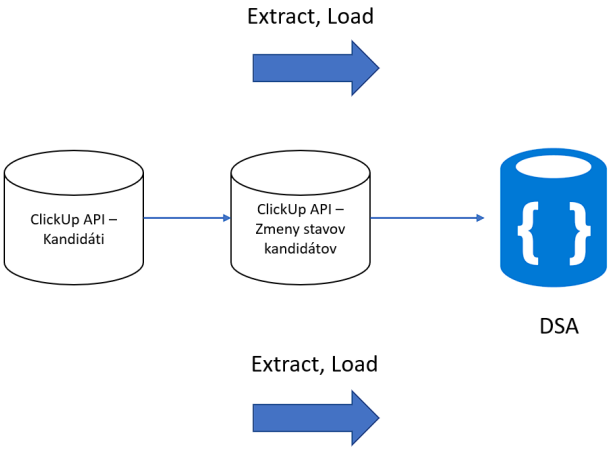



**Obrázok č. 28:** ETL proces - naplnenie DSA dátami z CU (Vlastné spracovanie)

Druhý a tretí ETL proces sú obdobné. Majú za úlohu extrahovať a načítať dáta z API o objednávkach a kandidátoch do DSA. Oba procesy sa skladajú z dvoch nadväzujúcich aktivít. Najprv je potrebné získať zoznam objednávok (kandidátov) a následne pre každú objednávku (kandidáta) extrahovať históriu stavov v príslušných procesoch.

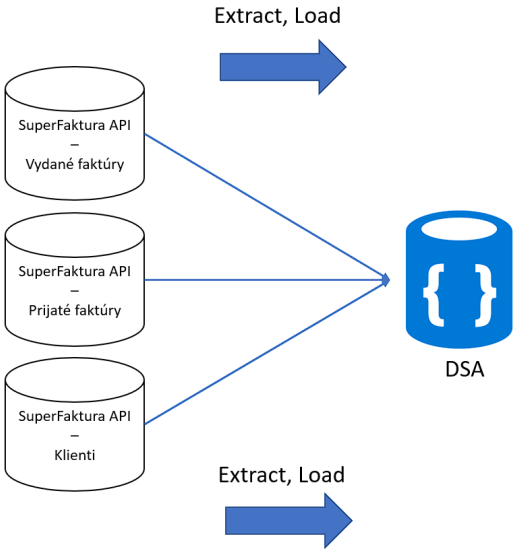



**Obrázok č. 29:** ETL proces - naplnenie DSA dátami o objednávkach (Vlastné spracovanie)



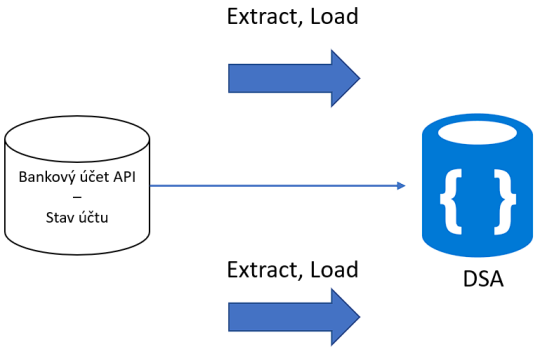

**Obrázok č. 30:** ETL proces - naplnenie DSA dátami o kandidátoch (Vlastné spracovanie)

Štvrtý ETL proces extrahuje dáta z API rozhrania systému SuperFaktura. Extrahujú sa dáta o vydaných faktúrach, prijatých faktúrach a klientoch. API rozhranie systému poskytuje dáta vo formáte JSON, takže je ich možne surovo uložiť do nerelačnej databáze ako dokumenty.




**Obrázok č. 31:** ETL proces - naplnenie DSA dátami zo SuperFaktury (Vlastné spracovanie)

Posledný ETL proces vo fáze naplnenia dočasného úložiska dát má za úlohy pomocou API rozhrania bankového účtu extrahovať informáciu o stave bankového účtu a uložiť ju do DSA.



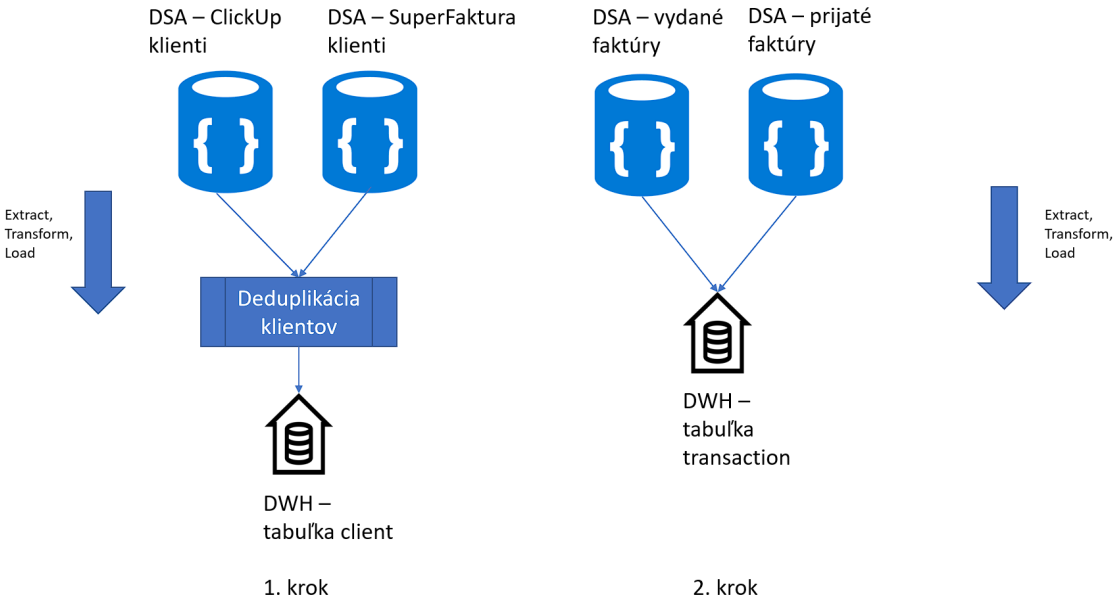

**Obrázok č. 32:** ETL proces - naplnenie DSA dátami z bank. účtu (Vlastné spracovanie)

Po ukončení predošlých ETL procesov sa v dočasnom úložisku dát (DSA) nachádzajú všetky potrebné dáta, ktoré sa budú integrovať do dátového skladu.

#### 3.3.2.2 Naplnenie dátového skladu

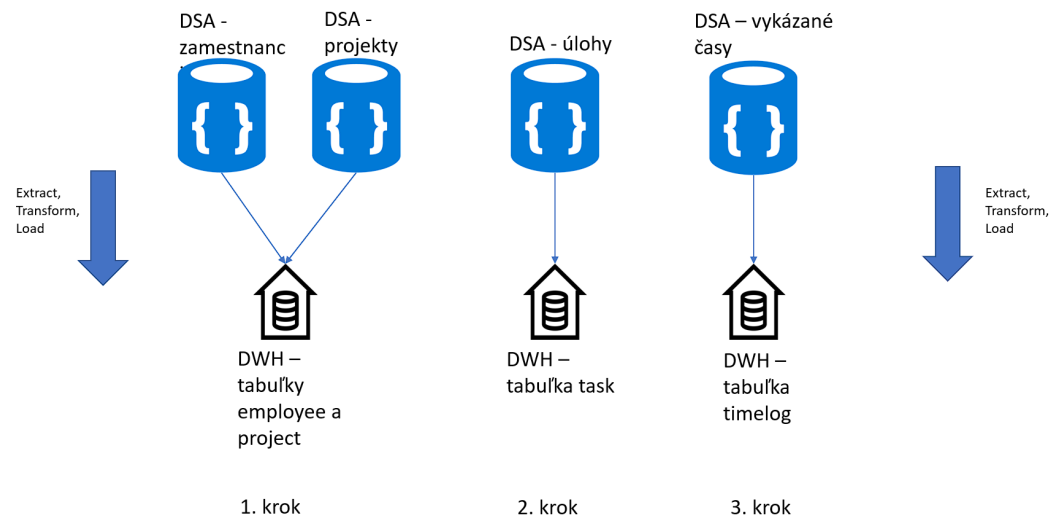
Prvý ETL proces vo fáze naplnenia dátového skladu má 2 na sebe nadväzujúce aktivity. Prvou aktivitou je načítanie klientov z DSA z oboch dátových zdrojov ClickUp a

SuperFaktura a deduplikovanie rovnakých záznamov pred načítaním dát do dátového skladu - tabuľky client. Druhou aktivitou je načítanie vydaných a prijatých faktúr z DSA a uloženie dát do tabuľky transaction. V oboch aktivitách prebieha transformácia dát do požadovaného formátu.



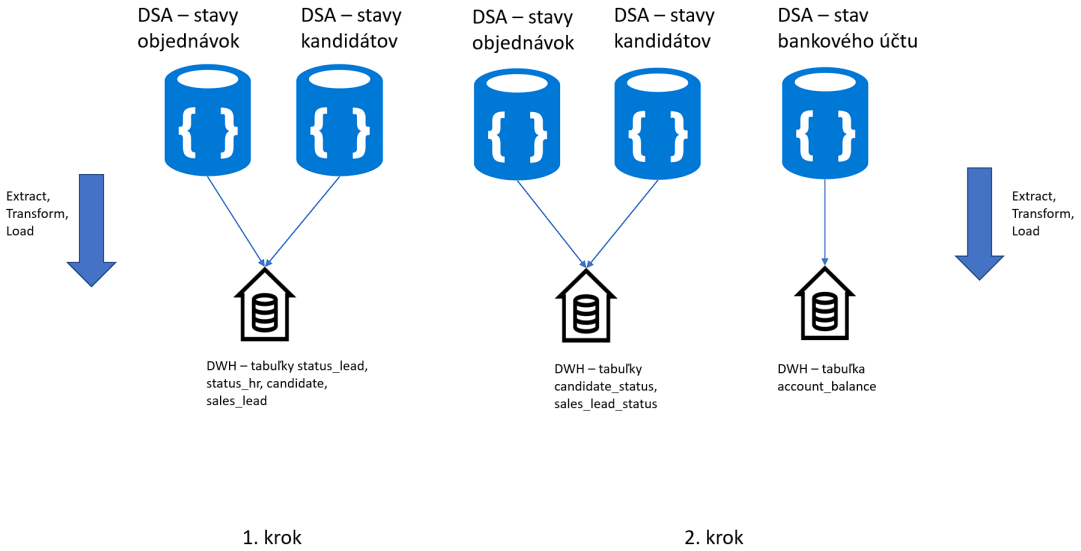

**Obrázok č. 33:** ETL proces - naplnenie dátového skladu dátami o klientoch a faktúrach (Vlastné spracovanie)

Druhý ETL proces vo fáze naplnenia dátového skladu dátami z dočasného úložiska dát má za úlohu extrahovať, transformovať a načítať dáta o zamestnancov, projektoch, úlohách a vykázaných časov. ETL proces má 3 nadväzujúce a závislé kroky, pretože úlohy sú logicky viazané na projekty, vykázané časy sú viazané na úlohy a projekty na klientov. Bez zachovania týchto krokov by sa pri načítaní dát do dátového skladu stratili väzby medzi záznamami.




**Obrázok č. 34:** ETL proces - naplnenie dátového skladu dátami o vykázaných časoch (Vlastné spracovanie)

Posledný tretí ETL proces vo fáze naplnenia dátového skladu integruje posledné zvyšné dáta do dátového skladu. Prvým krokom je načítať všetky stavy objednávok a kandidátov z DSA a transformovať dáta do príslušných tabuliek. Následne je možné vytvoriť tabuľky faktov pre objednávky a kandidátov - táto nadväznosť sa musí rešpektovať pre zachovanie väzieb medzi tabuľkami. Posledne je možné načítať dáta o stave bankového účtu.




**Obrázok č. 35:** ETL proces - naplnenie dátového skladu dátami o objednávkach, kandidátoch a stave bank. účtu (Vlastné spracovanie)

Na konci každého ETL procesu je potrebné obsah DSA databáze vymazať.

### 3.3.3 Implementácia ETL procesov

Táto podkapitola obsahuje ukážky Python skriptov, akým spôsobom sú ETL procesy implementované.

#### 3.3.3.1 Načítanie dát do DSA

V rámci ETL procesov vo fáze naplnenia DSA existujú aktivity, ktoré nie sú závislé na žiadnej inej aktivite. K týmto aktivitám patria:

- načítanie dát z API systému ClickUp - zamestnancov, vykázaných časov, projektov, úloh, klientov
- načítanie dát z API systému SuperFaktura - vydaných faktúr, prijatých faktúr, klientov
- načítanie dát z API bankového účtu

Nasledujúci kód ukazuje spôsob načítania a naplnenia dát o zamestnancoch zo systému ClickUp do DSA - nerelačnej databáze MongoDB. Rovnakým spôsobom sú načítané všetky ostatné dáta v nezávislých aktivitách spojené so systémom ClickUp, SuperFaktura a stavom bankového účtu. Jediný rozdiel je v URL smerujúcich API dotazov. Pre každú aktivitu existuje príslušný skript, ktorá kopíruje JSON objekty prvkov zo zdrojových dát do DSA. Z toho dôvodu sa uvádza len jeden kód.

```
headers = {
  'Authorization': config['clickup_apiKey'],
  'Content-Type': 'application/json; charset=UTF-8'
}

myclient = pymongo.MongoClient("mongodb://"+config['mongoDb']['user']['name']+":"+config['mongoDb']['user']['pw']+"@"+config['mongoDb']['host']+"/")
mydb_clickup = myclient["clickUp"]
mycol_employees = mydb_clickup['employees']

request = Request('https://api.clickup.com/api/v2/list/XXXXX/task?include_closed=true', headers=headers)
members = []
response_body = urlopen(request).read()
response_body_json = json.loads(response_body)
members = response_body_json['tasks']

for member in members:
    member['_id'] = member['assignees'][0]['id'] if len(member['assignees']) > 0 else member['id']
    update_objects_members.append(pymongo.ReplaceOne( {'_id': member['_id']},  member, upsert=True))
    if len(update_objects_members) == 1000:
        mycol_employees.bulk_write(update_objects_members)
        update_objects_members = []
if len(update_objects_members) > 0:
    mycol_employees.bulk_write(update_objects_members)
```

**Obrázok č. 36:** Načítanie dát nezávislými ETL aktivitami do DSA (Vlastné spracovanie)

Ďalším typom ETL aktivít vo fáze extrakcie dát do DSA sú závislé aktivity. Tieto závislé aktivity extrahujú dáta o objednávkach a kandidátoch dvoma závislými ETL aktivitami. Takéto nastavenie ETL procesov je z objektívnych dôvodov fungovania API rozhrania systému ClickUp. V prvej časti je potrebné zistiť všetky objednávky (alebo príslušných kandidátov), v druhej časti je potrebné iteratívne extrahovať históriu zmien pre každý jeden objekt (objednávku alebo kandidáta). Tento proces pre extrahovanie dát o objednávkach je implementovaný na nasledujúcom obrázku. Kód pre extrahovanie dát o obchodníkoch je založený na rovnakom princípe, z toho dôvodu sa tu neuvádza.

```
myclient = pymongo.MongoClient("mongodb://"+config['mongoDb']['user']['name']+":"+config['mongoDb']['user']['pw']+"@"+config['mongoDb']['host']+"/")
mydb_clickup = myclient["clickUp"]
mycol_obchod = mydb_clickup["obchod"]
mycol_obchod_statusy = mydb_clickup['obchod_statusy']
obchod = []
length = 100
page=0
while length == 100:
    request = Request('https://api.clickup.com/api/v2/list/XXXXX/task?include_closed=true&date_updated_gt='+str(date_from_ts)+'&page='+str(page), headers=headers)
    response_body = urlopen(request).read()
    response_body = json.loads(response_body)
    obchod.extend(response_body['tasks'])
    length = len(response_body['tasks'])
    page = page + 1
if len(obchod) > 0:
    query_string = update_object = obchod_data = []
    for lead in obchod:
        won = hourRate = offerMax = offerMin = offerFinal = designerAmount = analystAmount = None
        custom_data = {'id': lead['id'], 'name': lead['name']}
        if lead['status']['status'] == 'won':
            won = True
        elif lead['status']['status'] == 'lost':
            won = False
        for x in lead['custom_fields']:
            if x['name'] == 'Hodinovka Bez DPH' and 'value' in x:
                hourRate = float(x['value'])
            if x['name'] == 'Horní hranice' and 'value' in x:
                offerMax = float(x['value'])
            if x['name'] == 'Spodní hranice' and 'value' in x:
                offerMin = float(x['value'])
            if x['name'] == 'částka/budget v kč' and 'value' in x:
                offerFinal = float(x['value'])
            if x['name'] == 'Počet designerů' and 'value' in x:
                designerAmount = int(x['value'])
            if x['name'] == 'Počet analytiků' and 'value' in x:
                analystAmount = int(x['value'])
        custom_data['won'] = won
        custom_data['hourRate'] = hourRate
        custom_data['offerMax'] = offerMax
        custom_data['offerMin'] = offerMin
        custom_data['offerFinal'] = offerFinal
        custom_data['designerAmount'] = designerAmount
        custom_data['analystAmount'] = analystAmount
        obchod_data.append(custom_data)
        query_string.append('task_ids='+str(lead['id']))
        lead['_id'] = lead['id']
        update_object.append(pymongo.ReplaceOne( {'_id': lead['_id']},  lead, upsert=True))
        if len(update_object) == 1000:
            mycol_obchod.bulk_write(update_object)
            update_object = []
    if len(update_object) > 0:
        mycol_obchod.bulk_write(update_object)
        update_object = []
    def chunkify(lst,n):
        return [lst[i::n] for i in range(n)]
    tenths = int(len(query_string)/10)
    help_query = chunkify(query_string, tenths+1)
    obchod_statusy = []
    for x in help_query:
        string = '&'.join(x)
        request = Request('https://api.clickup.com/api/v2/task/bulk_time_in_status/task_ids/?'+str(string), headers=headers)
        response_body = urlopen(request).read()
        response_body = json.loads(response_body)
        obchod_statusy.append(response_body)
    status_data = []
    for node in obchod_statusy:
        for key, value in node.items():
            status_data.append({'_id': key, 'id': key, 'status': value})
    for status in status_data:
        update_object.append(pymongo.ReplaceOne( {'_id': status['_id']},  status, upsert=True))
        if len(update_object) == 1000:
            mycol_obchod_statusy.bulk_write(update_object)
            update_object = []
    if len(update_object) > 0:
        mycol_obchod_statusy.bulk_write(update_object)
        update_object = []
```

**Obrázok č. 37:** Načítanie dát závislých ETL aktivít o obchodných dátach do DSA (Vlastné spracovanie)

### 3.3.3.1 Načítanie dát do dátového skladu

Po ukončení fáze nahrávania dát do DSA je možné dáta ETL procesmi extrahovať, transformovať a načítať do koncového dátového skladu. V tejto fáze je potrebné rešpektovať postupnosť navrhovaných krokov v rámci návrhu ETL, pre zachovanie relácií záznamov v koncových tabuľkách. Nasledujúci kód zobrazuje spôsob

extrahovania dát z nerelačnej databáze, transformáciu dát do potrebnej podoby a konečné načítanie transformovaných dát do príslušných relačných tabuliek. Pracuje s dátami o projektových úlohách a vykázanými časmi zamestnancov. Na prepojenie záznamov naprieč tabuľky (vytvorenie vzťahov) sú použité unikátne identifikátory objektov zo zdrojových systémov. Týmto spôsobom sú extrahované, transformované a načítané dáta do dátového skladu o zamestnancoch, projektoch, úlohách, vykázaných časoch, faktúrach a stave bankového účtu z DSA.

```
myclient = pymongo.MongoClient("mongodb://"+config['mongoDb']['user']['name']+":"+config['mongoDb']['user']['pw']+"@"+config['mongoDb']['host']+"/")
mydb_clickup = myclient["clickUp"]
mycol_time = mydb_clickup["time_entries"]
mycol_tasks = mydb_clickup["tasks"]

engine = create_engine('postgresql://'+str(config['postgre']['user']['name'])+':'+
                       str(config['postgre']['user']['pw'])+'@'+str(config['postgre']['host'])+'/'+config['postgre']['db'])
conn = engine.connect()

timeLogs = mycol_time.find({'start':{'$gte': date_from_ts}, 'end':{'$lte': current_time_ts}},
                           {'_id':1,'billable':1,'user.id':1,'task.id':1,'start':1,'end':1,'duration':1,'description':1,'tags.name':1,
                            'source':1,'at':1,'update_timestamp_ms':1,'was_removed':1,'task.id':1})
timeLogs_data = []
for x in timeLogs:
    timeLogs_data.append({'id': str(x['_id']),
                          'isBillable': x['billable'],
                          'employeeId': str(x['user']['id']),
                          'description': x['description'],
                          'tags': x['tags'][0]['name'] if len(x['tags']) > 0 else '',
                          'startDate': datetime.fromtimestamp(int(x['start'])/1000.0),
                          'endDate': datetime.fromtimestamp(int(x['end'])/1000.0),
                          'updatedDate': datetime.fromtimestamp(int(x['at'])/1000.0),
                          'durationMs': x['duration'],
                          'taskId': str(x['task']['id']) if 'task' in x else None,
                          'removed': x['was_removed'] if 'was_removed' in x else "false"
                          })

tasks = mycol_tasks.find({},{'_id':1, 'name': 1, 'tags.name':1,'parent':1})
tasks_data = []
for x in tasks:
    data_object ={'id': str(x['_id']),
                  'name': x['name'],
                  'tags': []}
    for y in x['tags']:
        data_object['tags'].append(y['name'])

    tasks_data.append(data_object)
```

**Obrázok č. 38:** Extrahovanie a transformácia dát z DSA (Vlastné spracovanie)

```
insert_stmt = insert(Task).values(tasks_data)

on_duplicate_key_stmt = insert_stmt.on_conflict_do_update(
    constraint = 'task_pkey',
    set_=dict(
        name = insert_stmt.excluded.name,
        tags = insert_stmt.excluded.tags,
        projectId = insert_stmt.excluded.projectId
    )
)

conn.execute(on_duplicate_key_stmt)

insert_stmt = insert(TimeLog).values(timeLogs_data)

on_duplicate_key_stmt = insert_stmt.on_conflict_do_update(
    constraint = 'timelog_pkey',
    set_=dict(
        employeeId = insert_stmt.excluded.employeeId,
        description = insert_stmt.excluded.description,
        tags = insert_stmt.excluded.tags,
        startDate = insert_stmt.excluded.startDate,
        endDate = insert_stmt.excluded.endDate,
        updatedDate = insert_stmt.excluded.updatedDate,
        durationMs = insert_stmt.excluded.durationMs,
        isBillable = insert_stmt.excluded.isBillable,
        taskId = insert_stmt.excluded.taskId,
        removed = insert_stmt.excluded.removed
    )
)

conn.execute(on_duplicate_key_stmt)
```

**Obrázok č. 39:** Načítanie transformovaných dát z DSA do DWH (Vlastné spracovanie)

Špeciálnym typom ETL procesu z DSA do dátového skladu je načítanie dát o objednávkach a kandidátoch. V rámci neho je potrebné z DSA načítať jednotlivé stavy objednávok (kandidátov) a extrahovať z týchto záznamov objekty týkajúce sa samotnej objednávky (kandidát), jednotlivé stavy týchto objektov a históriu stavov. Týmto je možné načítať dáta do príslušných 2 tabuliek dimenzií a jednej tabuľky faktov. Nasledujúci obrázok ukazuje implementáciu tohto procesu. Vzťahy medzi záznamami tabuliek sú zabezpečené pomocou unikátnych identifikátor objektov, ktoré ponúka API rozhrania systému ClickUp.

```
insert_stmt = insert(SalesLead).values(obchod_data)
on_duplicate_key_stmt = insert_stmt.on_conflict_do_update(
    index_elements = ['id'],
    set_=dict(
        name = insert_stmt.excluded.name,
        won = insert_stmt.excluded.won,
        hourRate = insert_stmt.excluded.hourRate,
        offerMax = insert_stmt.excluded.offerMax,
        offerMin = insert_stmt.excluded.offerMin,
        offerFinal = insert_stmt.excluded.offerFinal,
        designerAmount = insert_stmt.excluded.designerAmount,
        analystAmount = insert_stmt.excluded.analystAmount
    )
)
conn.execute(on_duplicate_key_stmt)
def get_sales_status_id(conn,name):
    query = conn.execute(text('select * from "status_sales" where "name" = \'' +name.lower()+ '\''))
    first_result = query.first()
    if first_result is not None:
        first_result = dict(first_result._mapping.items())
        return first_result['id']
    else:
        insert_stmt = insert(SalesStatus).values({
            'name': name.lower()
        })
        result = conn.execute(insert_stmt)
        return result.lastrowid
status_data
lead_status_data = []
for status in status_data:
    newest_status = {'index': -1, 'ts':0}
    index = 0
    for history in status['status']['status_history']:
        if len(status['status']['status_history']) > 2 and (history['status'] == 'won' or history['status'] == 'lost'):
            continue
        if int(history['total_time']['since']) > newest_status['ts']:
            newest_status['index'] = index
            newest_status['ts'] = int(history['total_time']['since'])
        index= index+1
    index = 0
    for history in status['status']['status_history']:
        status_id = get_sales_status_id(conn, history['status'])
        custom_data = {'leadId': status['id'],
                       'statusId': status_id,
                       'duration': int(history['total_time']['by_minute']),
                       'date_since': datetime.datetime.fromtimestamp(int(int(history['total_time']['since'])/1000))}
        if index == newest_status['index']:
            custom_data['is_last_status'] = True
        else:
            custom_data['is_last_status'] = False
        index = index+1
        lead_status_data.append(custom_data)
insert_stmt = insert(LeadSalesStatus).values(lead_status_data)
on_duplicate_key_stmt = insert_stmt.on_conflict_do_update(
    constraint = 'unique_lead_sales_status',
    set_=dict(
        duration = insert_stmt.excluded.duration,
        date_since = insert_stmt.excluded.date_since,
        is_last_status = insert_stmt.excluded.is_last_status
    )
)
conn.execute(on_duplicate_key_stmt)
```

**Obrázok č. 40:** Načítanie dát o objednávkach (kandidátoch) z DSA do DWH (Vlastné spracovanie)

#### 3.3.3.2 Deduplikácia dát

Ďalším špeciálnym typom ETL procesu je načítanie klientov, deduplikácia záznamov alebo prípadné spojenie záznamov do jedného záznamu. V rámci rôznych dátový zdrojov, ktoré uchováva rovnaké typy dát (zoznam klientov), sa môže stať, že rovnaký klient má odlišné spôsoby záznamu v dátových zdrojov. Klient *ABC*, ktorého legálny názov je *ABC s.r.o.*, môže byť rôznymi spôsobmi uložený. Je možné ho uložiť ako:

- ABC
- ABC s.r.o.
- ABC s r o
- ABC s.r.o
- abc

Pričom v dátovom sklade, ktorý má byť jednotný zdroj pravdy, je ideálne, aby tento klient sa nachádzal v záznamoch klientov iba raz. Naviac môže sa stať, že jeden dátový zdroj obsahuje určité informácie o zázname, napríklad IČO, DIČ a adresu, a druhý dátový zdroj obsahuje číslo bankové účtu, IČO a primárny kontakt (napríklad emailovú adresu). V takom prípade je ideálne tieto dve záznamy spojiť do jedného záznamu dostať tak jediný záznam, ktorý nesie všetky informácie. Vybrané spôsoby, ktorým sa spájajú záznamy klientov v navrhovanom ETL procesu, sú:

- spojenie na základe rovnakého DIČ
- spojenie na základe rovnakého IČO
- spojenie na základe rovnakého číslu účtu
- spojenie na základe rovnakej domény
- spojenie na základe rovnakého štandardizovaného mena.

Tento proces deduplikácie dát je prezentovaný na nasledujúcich troch obrázkoch kódov.

```
def client_merger(id, name, bankAccount, country, vatId, taxId, payTime, riskLevel,
                  domain, sourceType, sourceDataset, conn):
    id = 'empty' if id == None or not id else id
    name = 'empty' if name == None or not name else name
    bankAccount = 'empty' if bankAccount == None or not bankAccount else bankAccount
    vatId = 'empty' if vatId == None or not vatId else vatId
    taxId = 'empty' if taxId == None or not taxId else taxId
    domain = 'empty' if domain == None or not domain else domain
    datasourceID = get_data_source_id(conn, sourceType.lower(), sourceDataset.lower())
    name = name.lower()
    name = re.sub(r'\s+(s\.?\s*r\.?\s*(o\.)?|sro|sr|a\.s\.|as|z\.?s\.?)$', '', name) # company types
    name = re.sub(r'(^ing\.?|bc\.?|,|\.(com|cz|de|net))', '', name) # remove ing., comma, domain
    name = re.sub(r'\s+', ' ', name)
    name = name.strip()
    name = name.replace('"', '')
    name = name.replace("'", '')
    domain = domain.lower()
    if re.search(r'gmail|seznam|email\.cz', domain):
        domain = 'empty'
    else:
        domain = re.sub(r'^.*@', '', domain)
    results = conn.execute(text('select * from datasource_row where "datasourceRowId" = \''+
                                +str(id) + '\' and "datasourceId" = \'' + str(datasourceID)+'\''))
    result_first = results.first()
    if result_first is not None:
        result_first = dict(result_first._mapping.items())
        client_db_table = result_first['table']
        client_db_rowId = result_first['tableRowId']
        client_db = conn.execute(text('select * from "' +client_db_table+ '" where id = '+client_db_rowId))
        client_db = client_db.first()
        if client_db is None:
            conn.execute(text('delete from "' +client_db_table+ '" where id = '+client_db_rowId))
        client_db = dict(client_db._mapping.items())
        name = None if name == 'empty' else name
        bankAccount = None if bankAccount == 'empty' else bankAccount
        vatId = None if vatId == 'empty' else vatId
        taxId = None if taxId == 'empty' else taxId
        domain = None if domain == 'empty' else domain
        client_data = {'id': client_db['id'],
            'name': client_db['name'] if client_db['name'] else name,
            'bankAccount': client_db['bankAccount'] if client_db['bankAccount'] else bankAccount,
            'country': client_db['country'] if client_db['country'] else country,
            'vatId': client_db['vatId'] if client_db['vatId'] else vatId,
            'taxId': client_db['taxId'] if client_db['taxId'] else taxId,
            'payTime': client_db['payTime'] if client_db['payTime'] else payTime,
            'riskLevel': client_db['riskLevel'] if client_db['riskLevel'] else riskLevel,
            'domain': client_db['domain'] if client_db['domain'] else domain  }
        insert_stmt = insert(Client).values(client_data)
        on_duplicate_key_stmt = insert_stmt.on_conflict_do_update(
            constraint = 'client_pkey',
            set_=dict(name=insert_stmt.excluded.name,
            bankAccount=insert_stmt.excluded.bankAccount,
            country=insert_stmt.excluded.country,
            vatId=insert_stmt.excluded.vatId,
            taxId=insert_stmt.excluded.taxId,
            payTime=insert_stmt.excluded.payTime,
            riskLevel=insert_stmt.excluded.riskLevel,
            domain=insert_stmt.excluded.domain))
        conn.execute(on_duplicate_key_stmt)
```

**Obrázok č. 41:** Deduplikačná funkcia klientov časť 1 (Vlastné spracovanie)

```
    insert_stmt = insert(DatasourceRow).values({'datasourceId': datasourceID,
        'datasourceRowId': id,
        'lastUpdatedDate': datetime.today()})
    on_duplicate_key_stmt = insert_stmt.on_conflict_do_update(constraint = 'unique_datasourceId',
        set_=dict(
            datasourceId= insert_stmt.excluded.datasourceId,
            datasourceRowId= insert_stmt.excluded.datasourceRowId,
            lastUpdatedDate= insert_stmt.excluded.lastUpdatedDate))
    conn.execute(on_duplicate_key_stmt)
else:
    matchTags = [name,taxId,vatId,bankAccount,domain]
    ##0-name, 1-taxId, 2 - vatId, 3 - bankAccount, 4 - domain
    for x in range(5):
        if x==0 and name != 'empty':
          client_db = conn.execute(text('select * from "client" where "name" = \''+str(name)+'\''))
          client_db_first = client_db.first()
        elif x==1 and taxId != 'empty':
          client_db = conn.execute(text('select * from "client" where "taxId" = \''+str(taxId)+'\''))
          client_db_first = client_db.first()
        elif x==2 and vatId != 'empty':
          client_db = conn.execute(text('select * from "client" where "vatId" = \''+str(vatId)+'\''))
          client_db_first = client_db.first()
        elif x==3 and bankAccount != 'empty':
          client_db = conn.execute(text('select * from "client" where "bankAccount" = \''+str(bankAccount)+'\''))
          client_db_first = client_db.first()
        elif x==4 and domain != 'empty':
          client_db = conn.execute(text('select * from "client" where "domain" = \''+str(domain)+'\''))
          client_db_first = client_db.first()
        else:
            client_db_first = None
        if client_db_first is not None:
            break
    if client_db_first is None:
        name = None if name == 'empty' else name
        bankAccount = None if bankAccount == 'empty' else bankAccount
        vatId = None if vatId == 'empty' else vatId
        taxId = None if taxId == 'empty' else taxId
        domain = None if domain == 'empty' else domain
        client_data = {'name': name,
            'bankAccount': bankAccount,
            'country': country,
            'vatId': vatId,
            'taxId': taxId,
            'payTime': payTime,
            'riskLevel': riskLevel,
            'domain': domain}
        insert_stmt = insert(Client).values(client_data)
        result = conn.execute(insert_stmt)
        inserted_client_id = result.lastrowid
        actual_time = datetime.today()
        datasourcerow_data = {'datasourceId': datasourceID,
            'datasourceRowId': id,
            'table': 'client',
            'tableRowId': inserted_client_id,
            'date': actual_time,
            'lastUpdatedDate': actual_time}
        insert_stmt = insert(DatasourceRow).values(datasourcerow_data)
        result = conn.execute(insert_stmt)
```

**Obrázok č. 42:** Deduplikačná funkcia klientov časť 2 (Vlastné spracovanie)

```
    else:
        client_result_first = dict(client_db_first._mapping.items())
        name = None if name == 'empty' else name
        bankAccount = None if bankAccount == 'empty' else bankAccount
        vatId = None if vatId == 'empty' else vatId
        taxId = None if taxId == 'empty' else taxId
        domain = None if domain == 'empty' else domain
        client_data = {'id': client_result_first['id'],
            'name': client_result_first['name'] if client_result_first['name'] else name,
            'bankAccount': client_result_first['bankAccount'] if client_result_first['bankAccount'] else bankAccount,
            'country': client_result_first['country'] if client_result_first['country'] else country,
            'vatId': client_result_first['vatId'] if client_result_first['vatId'] else vatId,
            'taxId': client_result_first['taxId'] if client_result_first['taxId'] else taxId,
            'payTime': client_result_first['payTime'] if client_result_first['payTime'] else payTime,
            'riskLevel': client_result_first['riskLevel'] if client_result_first['riskLevel'] else riskLevel,
            'domain': client_result_first['domain'] if client_result_first['domain'] else domain}
        insert_stmt = insert(Client).values(client_data)
        on_duplicate_key_stmt = insert_stmt.on_conflict_do_update(
            constraint = 'client_pkey',
            set_=dict(name=insert_stmt.excluded.name,
                bankAccount=insert_stmt.excluded.bankAccount,
                country=insert_stmt.excluded.country,
                vatId=insert_stmt.excluded.vatId,
                taxId=insert_stmt.excluded.taxId,
                payTime=insert_stmt.excluded.payTime,
                riskLevel=insert_stmt.excluded.riskLevel,
                domain=insert_stmt.excluded.domain))
        conn.execute(on_duplicate_key_stmt)
        actual_time = datetime.today()
        insert_stmt = insert(DatasourceRow).values({'datasourceId': datasourceID,
            'datasourceRowId': id,
            'table': 'client',
            'tableRowId': client_result_first['id'],
            'date': actual_time,
            'lastUpdatedDate': actual_time})
        conn.execute(insert_stmt)

def get_data_source_id(conn,sourceType,sourceDataset):
    query = conn.execute(text('select * from "datasource" where "type" = \'' +sourceType.lower()+ '\' and "dataset" = \'' +sourceDataset.lower()+'\''))
    first_result = query.first()
    if first_result is not None:
        first_result = dict(first_result._mapping.items())
        return first_result['id']
    else:
        insert_stmt = insert(Datasource).values({
            'type': sourceType.lower(),
            'dataset': sourceDataset.lower(),
            'lastSyncDate': datetime.today(),
            'isActive': True
        })
        result = conn.execute(insert_stmt)
        return result.lastrowid
```

**Obrázok č. 43:** Deduplikačná funkcia klientov časť 3 (Vlastné spracovanie)

### 3.3.3.3 Vytvorenie pohľadov (views) pre reporting

Po implementovaní ETL procesov je dátový sklad naplnený potrebnými dátami pre reporting. Vzhľadom na malé množstvo dát nedáva zmysel vytvárať oddelené dátové tržiská určené pre reportovanie oddelených oblastí spoločnosti. Pre oddelenie dát v navrhovanom riešení slúžia databázové pohľady (views), ktoré čerpajú dáta z celopodnikového dátového skladu.

#### Pohľad na vykázané časy

Tento pohľad čerpá dáta z tabuliek timelog, project, task a employee. Schéma tabuľky je nasledovná:

```
CREATE VIEW "timelog_report" AS
    SELECT p.name AS "Project",
        ta.name AS "Task",
        e.name AS "Employee",
        e."primaryJob",
        t."startDate",
        t."endDate",
        t."isBillable",
        t."durationMs",
        t."updatedDate",
        t.description,
        t.removed,
        t.tags
    FROM timelog t
        LEFT JOIN task ta ON t."taskId" = ta.id
        LEFT JOIN project p ON ta."projectId" = p.id
        LEFT JOIN employee e ON t."employeeId" = e.id;
```

**Obrázok č. 44:** Pohľad na vykázané časy (Vlastné spracovanie)

Výsledná tabuľka vyzerá nasledovne (dáta v tabuľke sú vzorové):

| Project | Task | Employee | primaryJob | startDate | endDate | isBillable | durationMs | updatedDate | description | removed | tags |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Projekt A | Nastavení Google Analytics [párty kýbl] | Zamestnanec A | analytika | 11.1.2022 14:00 | 11.1.2022 16:05 | TRUE | 7528619 | 12.1.2022 12:26 | Call s klientem a zaslání úkolů | FALSE | NULL |
| Projekt A | Nastavení Google Analytics | Zamestnanec B | vedení | 11.1.2022 14:00 | 11.1.2022 15:30 | TRUE | 5400296 | 11.1.2022 15:43 | NULL | FALSE | NULL |
| Projekt A | Stránka Marketáka | Zamestnanec C | back-office | 12.1.2022 10:51 | 12.1.2022 10:58 | FALSE | 444956 | 12.1.2022 10:58 | NULL | FALSE | NULL |
| Projekt A | Nastavení Google Analytics | Zamestnanec A | analytika | 12.1.2022 9:45 | 12.1.2022 11:00 | TRUE | 4500000 | 12.1.2022 12:27 | Call k plánování prací | FALSE | NULL |
| Projekt A | Nastavení Google Analytics | Zamestnanec A | analytika | 12.1.2022 12:00 | 12.1.2022 12:28 | TRUE | 1680000 | 12.1.2022 12:28 | Plánování prací k nastavení + požadavky | FALSE | NULL |
| Projekt B | Lišta | Zamestnanec B | vedení | 12.1.2022 15:04 | 12.1.2022 15:33 | FALSE | 1722693 | 12.1.2022 15:34 | NULL | FALSE | NULL |
| Projekt C | upravení textů inzerátů | Zamestnanec C | back-office | 12.1.2022 18:25 | 12.1.2022 19:53 | FALSE | 5285122 | 12.1.2022 19:53 | NULL | FALSE | NULL |
| Projekt D | Lišta | Zamestnanec B | vedení | 12.1.2022 21:20 | 12.1.2022 21:36 | FALSE | 1011962 | 12.1.2022 21:36 | NULL | FALSE | NULL |
| Projekt E | Rozšířené měření konverzí | Zamestnanec B | vedení | 12.1.2022 22:09 | 12.1.2022 22:50 | FALSE | 2512874 | 12.1.2022 22:50 | NULL | FALSE | servis |
| Projekt E | Měření CJ | Zamestnanec B | vedení | 12.1.2022 23:27 | 13.1.2022 0:09 | FALSE | 2488702 | 13.1.2022 0:09 | NULL | FALSE | servis |
| Projekt E | Rozšířené měření konverzí | Zamestnanec B | vedení | 13.1.2022 0:16 | 13.1.2022 0:26 | FALSE | 557818 | 13.1.2022 0:26 | NULL | FALSE | servis |
| Projekt A | Nastavení Google Analytics | Zamestnanec A | analytika | 13.1.2022 8:45 | 13.1.2022 12:08 | FALSE | 12229580 | 13.1.2022 15:21 | Nastaveni GTM | FALSE | NULL |
| Projekt A | Nastavení Google Analytics | Zamestnanec A | analytika | 13.1.2022 12:41 | 13.1.2022 14:17 | FALSE | 5747292 | 13.1.2022 15:20 | Nastaveni GTM | FALSE | NULL |
| Projekt A | Nastavení Google Analytics | Zamestnanec A | analytika | 14.1.2022 7:15 | 14.1.2022 7:31 | FALSE | 949480 | 17.1.2022 7:37 | Nastavení GA | FALSE | NULL |
| Projekt A | Nastavení Google Analytics | Zamestnanec A | analytika | 17.1.2022 8:12 | 17.1.2022 9:20 | FALSE | 4067995 | 17.1.2022 9:48 | GA | FALSE | NULL |
| Projekt F | Měření GA4 na všech webech | Zamestnanec A | analytika | 17.1.2022 9:20 | 17.1.2022 9:48 | TRUE | 1706625 | 18.1.2022 11:15 | Konzultace projektu | FALSE | NULL |

**Obrázok č. 45:** Tabuľka časov (Vlastné spracovanie)

### Pohľad na faktúry

Pohľad spája dáta faktúr a klientov. Je vytvorená nasledovne:

```sql
CREATE VIEW "transactions_report" AS
    SELECT t.id,
        t.currency,
        t.amount,
        t."amount_with_VAT",
        t.vat,
        t.type,
        t.name,
        t."txCategory",
        t."txSubcategory",
        t.transaction_number,
        t.date,
        t.status,
        c.name AS "Client Name"
    FROM transactions t
        LEFT JOIN client c ON t."clientId" = c.id;
```

**Obrázok č. 46: Pohľad na faktúry (Vlastné spracovanie)**

Výsledná tabuľka vyzerá nasledovne (dáta v tabuľke sú vzorové).

| id | currency | amount | amount_with_VAT | vat | type | name | txCategory | txSubcategory | transaction_number | date | status | Client Name |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1000001 | CZK | 640 | 695 | 63.1800003051758 | invoice | faktura služba A | Kniha | NULL | 1000001 | 27.11.2017 0:00 | paid | klient A |
| 1000276 | CZK | 5900 | 7139 | 1239 | invoice | služba B | Školení | Strategická optimalizace | 1000276 | 27.11.2017 0:00 | paid | klient B |
| 1000466 | CZK | 5900 | 7139 | 1239 | invoice | faktura služba B | Školení | Strategická optimalizace | 1000466 | 27.11.2017 0:00 | paid | klient C |
| 1000472 | CZK | 5900 | 7139 | 1239 | invoice | služba C | Školení | Strategický výzkum | 1000472 | 27.11.2017 0:00 | paid | klient D |
| 1002233 | CZK | 640 | 695 | 63.1800003051758 | invoice | faktura služba A | Kniha | NULL | 1002233 | 28.11.2017 0:00 | paid | klient E |
| 1002952 | CZK | 5900 | 7139 | 1239 | invoice | služba A | Školení | Strategický výzkum | 1002952 | 28.11.2017 0:00 | paid | klient F |
| 1005993 | CZK | 640 | 695 | 63.1800003051758 | expense | faktura služba A | Kniha | NULL | 1005993 | 30.11.2017 0:00 | paid | klient G |
| 1008084 | CZK | 640 | 695 | 63.1800003051758 | invoice | faktura služba B | Kniha | NULL | 1008084 | 1.12.2017 0:00 | paid | klient H |
| 1008502 | CZK | 15000 | 18150 | 3150 | invoice | služba D | Projekt | NULL | 1008502 | 1.12.2017 0:00 | paid | klient I |
| 1009262 | CZK | 500 | 500 | 0 | expense | Náklad E | Nájemné | NULL | 1009262 | 2.12.2017 0:00 | paid | klinet J |
| 1009263 | CZK | 500 | 500 | 0 | invoice | faktura služba A | Nájemné | NULL | 1009263 | 2.12.2017 0:00 | paid | klient K |
| 1009264 | CZK | 500 | 500 | 0 | invoice | Náklad A | Nájemné | NULL | 1009264 | 2.12.2017 0:00 | paid | klient L |
| 1009265 | CZK | 500 | 500 | 0 | expense | Náklad A | Nájemné | NULL | 1009265 | 2.12.2017 0:00 | paid | klient M |
| 1009266 | CZK | 500 | 500 | 0 | invoice | Náklad D | Nájemné | NULL | 1009266 | 2.12.2017 0:00 | paid | klient N |
| 1009267 | CZK | 500 | 500 | 0 | invoice | Faktura služna C | Nájemné | NULL | 1009267 | 2.12.2017 0:00 | paid | klient O |
| 1009268 | CZK | 500 | 500 | 0 | invoice | Náklad B | Nájemné | NULL | 1009268 | 2.12.2017 0:00 | paid | klient P |
| 1010463 | CZK | 5900 | 7139 | 1239 | invoice | Faktura služna D | Školení | Strategická optimalizace | 1010463 | 4.12.2017 0:00 | paid | klient Q |

**Obrázok č. 47:** Tabuľka financií (Vlastné spracovanie)

### Pohľad na kandidátov

Tento pohľad spája tabuľky candidate, status_hr a candidate_status.

```
CREATE VIEW "hr_report" AS
    SELECT c.duration,
    c.date_since,
    c.is_last_status,
    s.name,
    s.accepted,
    s."position",
    s.status,
    s.note,
    hr.name AS "Cadidate_status"
   FROM candidate_status c
     LEFT JOIN candidate s ON c."candidateId" = s.id
     LEFT JOIN status_hr hr ON c."statusId" = hr.id;
```

**Obrázok č. 48:** Pohľad na stavy kandidátov (Vlastné spracovanie)

Výsledná tabuľka vyzerá nasledovne (dáta v tabuľke sú vzorové):

| duration | date_since | is_last_status | name | accepted | position | status | note | Cadidate_status |
|---|---|---|---|---|---|---|---|---|
| 40213 | 2022-06-09 15:35:40 | False | Kandidát A | True | analytik | NULL | NULL | open |
| 5307 | 2022-04-25 14:14:11 | False | Kandidát B | False | analytik | NULL | Po prvním pol | pohovor |
| 49862 | 2022-04-29 06:41:28 | True | Kandidát B | False | analytik | NULL | Po prvním pol | hr workshop |
| 146084 | 2022-06-02 21:44:05 | False | Kandidát B | False | analytik | NULL | Po prvním pol | archivovat |
| 6866 | 2022-04-20 19:48:02 | False | Kandidát B | False | analytik | NULL | Po prvním pol | open |
| 76277 | 2022-03-03 13:59:38 | True | Kandidát C | False | analytik | NULL | Na pohovovo | pohovor |
| 201251 | 2022-04-25 14:16:56 | False | Kandidát C | False | analytik | NULL | Na pohovovo | archivovat |
| 2328 | 2022-03-01 23:10:50 | False | Kandidát C | False | analytik | NULL | Na pohovovo | open |
| 14915 | 2022-08-12 15:01:49 | True | Kandidát D | False | designer | NULL | Aktuálně moc | pohovor |
| 5223 | 2022-08-08 23:57:59 | False | Kandidát D | False | designer | NULL | Aktuálně moc | open |
| 9866 | 2022-08-12 12:56:32 | False | Kandidát E | True | designer | NULL | NULL | pohovor |
| 9949 | 2022-08-19 09:23:16 | False | Kandidát E | True | designer | NULL | NULL | hr workshop |

**Obrázok č. 49:** Tabuľka kandidátov (Vlastné spracovanie)

### Pohľad na objednávky

Pohľad na objednávky spája tabuľky sales_lead, status_sales a sales_lead_status.

```sql
CREATE VIEW "sales_report" AS
    SELECT sl.name AS "Poptávka",
    sl.won,
    sl."hourRate",
    sl."offerMax",
    sl."offerMin",
    sl."offerFinal",
    sl."analystAmount",
    sl."designerAmount",
    ss.name,
    sls.duration,
    sls.date_since,
    sls.is_last_status
   FROM sales_lead_status sls
     LEFT JOIN sales_lead sl ON sls."leadId" = sl.id
     LEFT JOIN status_sales ss ON sls."statusId" = ss.id
  ORDER BY sl.name, sls.date_since;
```

**Obrázok č. 50:** Pohľad na objednávky (Vlastné spracovanie)

Výsledná tabuľka vyzerá nasledovne (dáta v tabuľke sú vzorové):

| Poptávka | won | hourRate | offerMax | offerMin | offerFinal | analystAmount | designerAmount | name | duration | date_since | is_last_status |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Projekt A | True | 500 | 200000 | 17600 | 44000 | 1 | 1 | lead | 264 | 26.9.2022 9:24 | FALSE |
| Projekt A | True | 500 | 200000 | 17600 | 44000 | 1 | 1 | proposal | 2507 | 26.9.2022 13:48 | FALSE |
| Projekt A | True | 500 | 200000 | 17600 | 44000 | 1 | 1 | offer accepted | 27730 | 28.9.2022 7:36 | TRUE |
| Projekt A | True | 500 | 200000 | 17600 | 44000 | 1 | 1 | won | 83457 | 17.10.2022 13:46 | FALSE |
| Projekt B | False | 500 | 1284800 | 642400 | NULL | 1 | 2 | lead | 1040 | 30.6.2022 12:28 | FALSE |
| Projekt B | False | 500 | 1284800 | 642400 | NULL | 1 | 2 | contact made | 188 | 1.7.2022 5:49 | FALSE |
| Projekt B | False | 500 | 1284800 | 642400 | NULL | 1 | 2 | discovery meeting | 10153 | 1.7.2022 8:57 | FALSE |
| Projekt B | False | 500 | 1284800 | 642400 | NULL | 1 | 2 | proposal | 9061 | 21.7.2022 8:31 | FALSE |
| Projekt B | False | 500 | 1284800 | 642400 | NULL | 1 | 2 | negotiation | 270314 | 21.7.2022 11:32 | TRUE |
| Projekt B | False | 500 | 1284800 | 642400 | NULL | 1 | 2 | lost | 9625 | 18.1.2023 10:26 | FALSE |

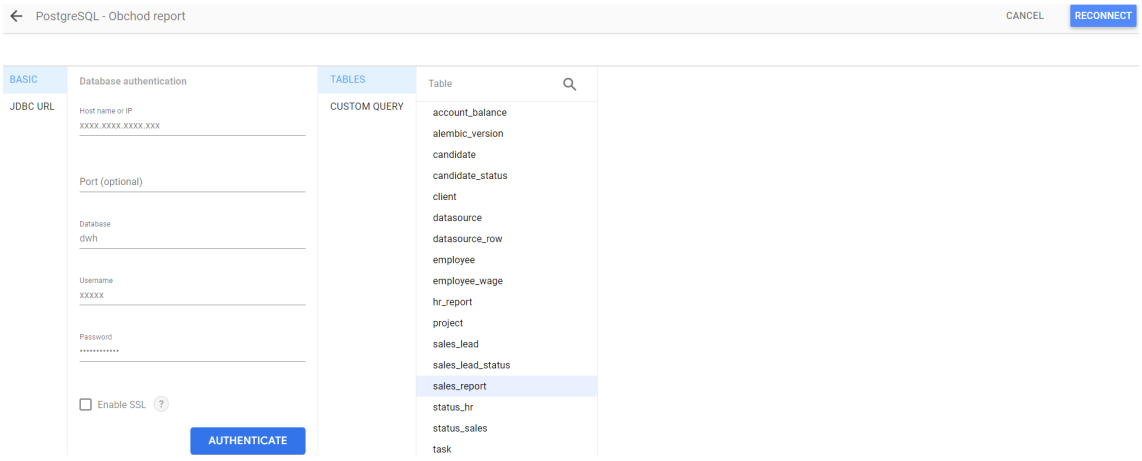
**Obrázok č. 51:** Tabuľka objednávok (Vlastné spracovanie)

#### 3.3.3.4 Načítanie dát do Looker Studia

Aby dáta vo vytvorených tabuľkách mohli byť vizualizované, je potrebné ich načítať do vizualizačného nástroja Looker Studio. Looker Studio obsahuje natívny konektor do PostgreSQL databáze, technológia, v ktorom beží navrhovaný dátový sklad.

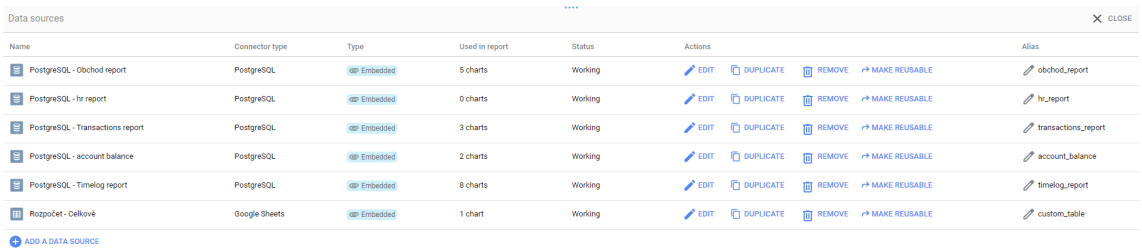
V konektoru je potrebné definovať prístupové údaje do databáze - doménu alebo IP adresu, port, názov databáze, prístupové údaje. Po pripojení sa zobrazia všetky tabuľky včetne vytvorených pohľadov (views).

Pre každú jednu tabuľku (databázový view) je potrebný pridať samostatný Resource (Zdroj) typu PostgreSQL s napojením na konkrétny pohľad (view). Nastavenie pripojenia na pohľad objednávok zobrazuje nasledujúci obrázok.




**Obrázok č. 52:** Nastavenie prepojenia Looker Studio s dátovým skladom (Vlastné spracovanie)

Pre každú potrebnú tabuľku alebo databázový pohľad (view) existuje samotný Dátový zdroj. Z týchto dátových zdrojov čerpajú vizualizácie v nástroje dáta.

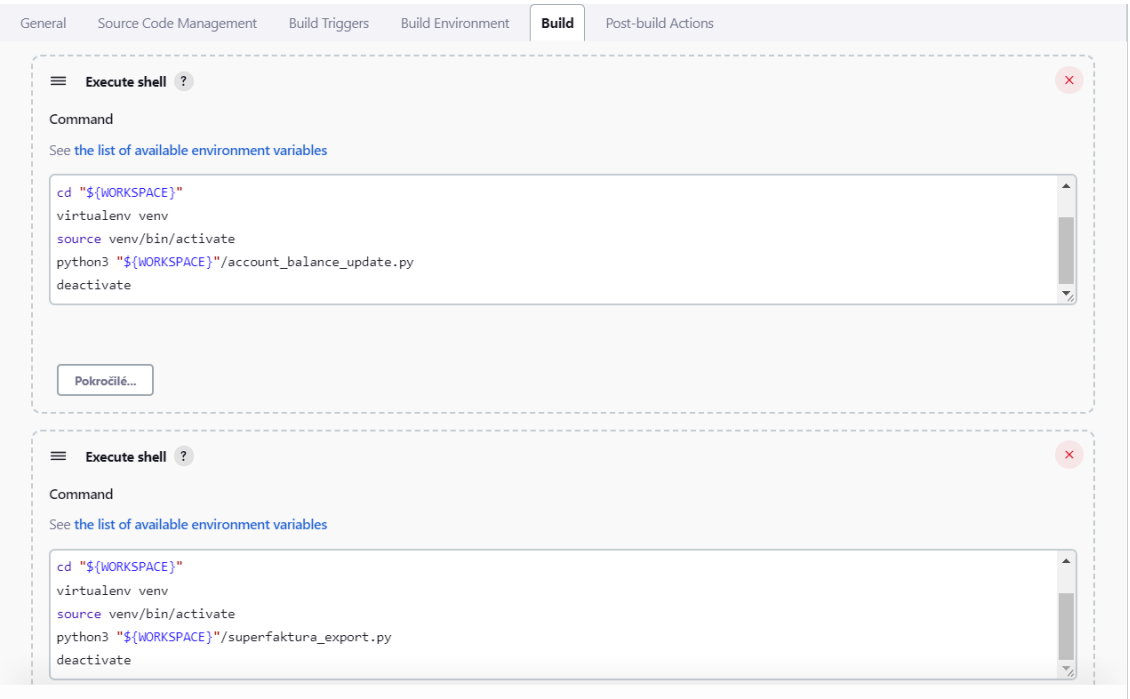



**Obrázok č. 53:** Zoznam zdrojov dát v Looker Studio (Vlastné spracovanie)

### 3.3.3.5 Automatizácia ETL procesov

Návrh nahrávania dát do DSA a dátového skladu pomocou ETL procesov je automatizovaný proces, ktorý nie je potrebné spúšťať manuálne. Automatizované spúšťanie Python skriptov s ETL procesmi je zabezpečené pomocou nástroja Jenkins.V rámci nástroja Jenkins je vytvorený jeden projekt, ktorý zahrňuje celý ETL proces. Spúšťanie ETL procesov je naplánované na 2 hodiny ráno každý deň. Správne poradie

spustenia závislých aktivít v rámci ETL procesov je riadené v projektu Jenkins jednotlivými "jobmi", ktoré spúšťajú príslušné skripty.




**Obrázok č. 54:** Automatizácia ETL skriptov v nástroji Jenkins (Vlastné spracovanie)

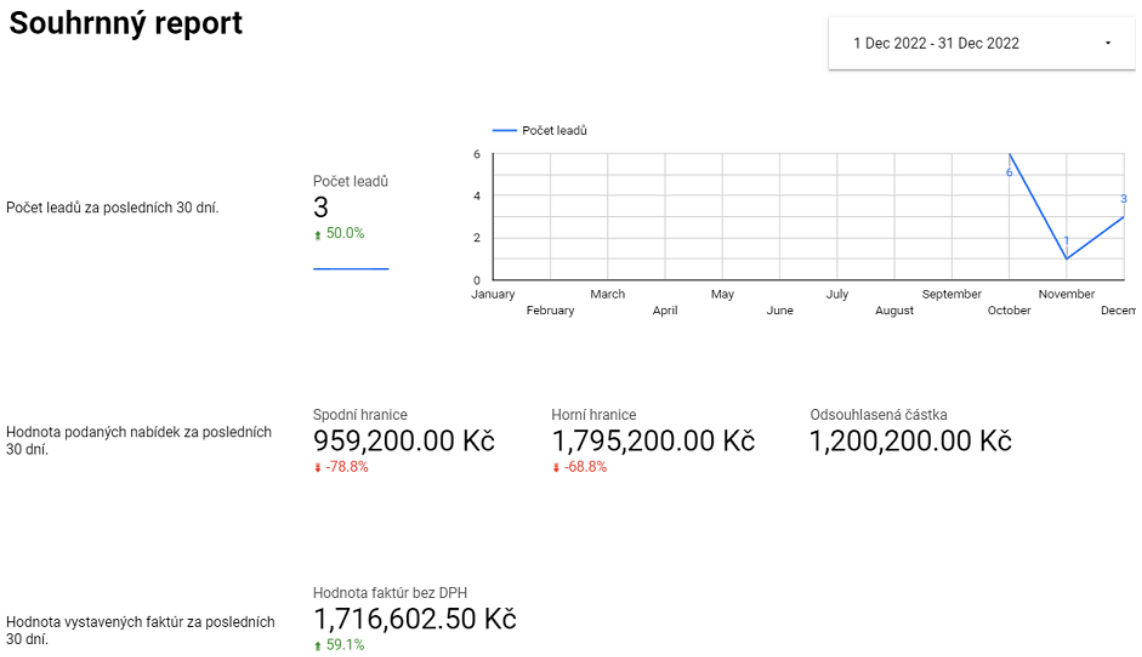
## 3.4 Reporting

Táto časť práce prezentuje implementované reporty v nástroji Looker Studio na základe informačných potrieb zadávateľa. Implementácia konkrétnej podoby reportov prebieha agilne, to znamená, že postupne sa iteruje na základe spolupráce a spätnej väzby od užívateľov reportov. Hlavnými užívateľmi nasledujúcich reportov je vedenie spoločnosti, skladajúci sa z CEO, CFO a COO. Výsledkom implementácie reportingu je 5 dashboardov. Vybraný design nasledujúcich reportov je jednoduchý, transparentný, z nekomplexnými vizualizáciami a pomocnými textami. V nasledujúcej časti podkapitoly sú predstavené implementované reporty. Čísla vo vizualizáciach sú vypočítané na základe cvičiacej dátovej sady, neukazujú výpočty z reálnych dát spoločnosti.

**Súhrnný report** je prvá stránka v rámci dashboardov. Report obsahuje súhrnné podnikové ukazovatele za posledných 30 dní. Obsahuje:

- počet nových obchodných objednávok, percentuálnu zmenu oproti predošlých 30 dní a ich vývoj v čase podľa mesiacov,
- celkovú hodnotu podaných ponúk rozdelených podľa jednotlivých hodnôt ponúk - dolná hranica ponuky, horná hranica ponuky a ich percentuálnu zmenu oproti predošlému intervalu,
- celkovú hodnotu úspešne vyobchodovaných projektov,
- celkovú hodnotu vystavených faktúr za posledných 30 dní a percentuálnu zmenu oproti predchádzajúcemu intervalu,
- filter pre zmenu prednastaveného reportovaného časového intervalu.

Nasledujúci obrázok zobrazuje stránku so súhrnným reportom.



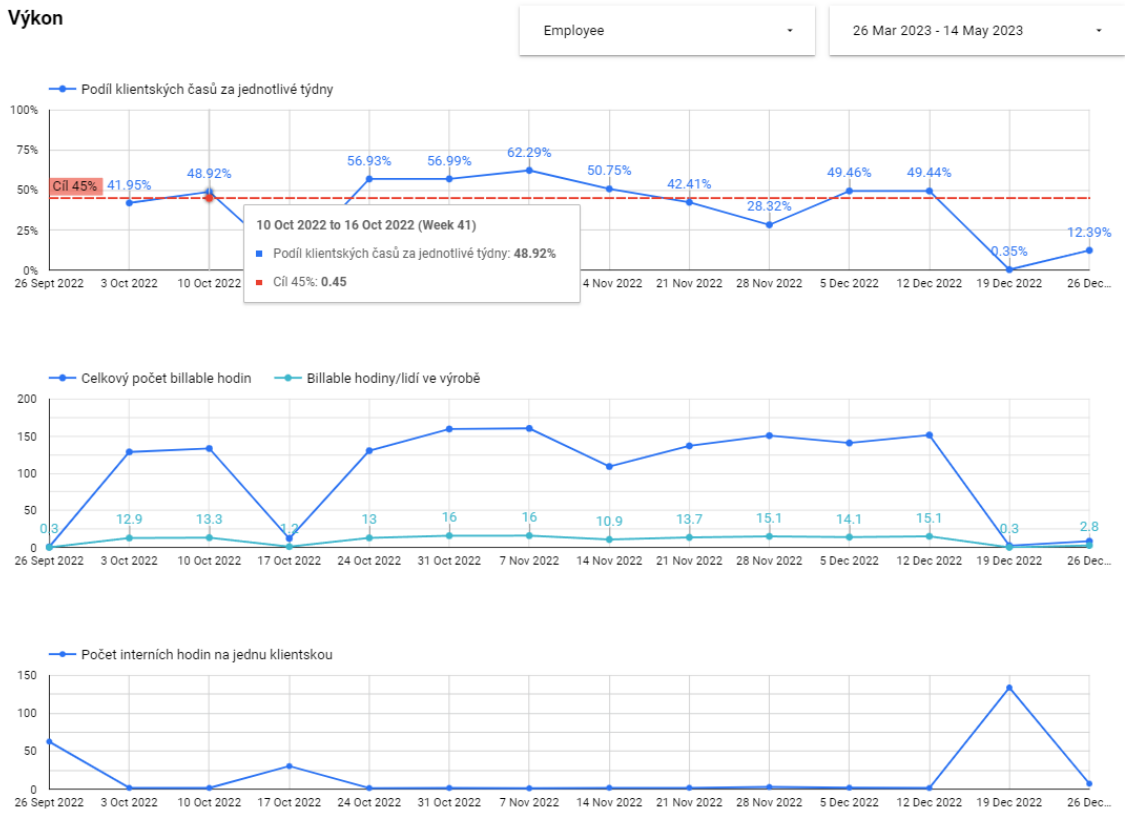

**Obrázok č. 55:** Súhrnný report (Vlastné spracovanie)

**Report o výkone výroby** zobrazuje dáta o vykázaných časoch ľudí vo výrobe (analytické a designové oddelenie). Obsahuje:

- graf s vývojom podielu faktúrovateľných vykázaných časov a celkového času zamestnancov vo výrobe, s vyznačeným cieľom 45 %, rozdelené na jednotlivé týždne v roku,

- celkový počet vykázaných faktúrovateľných hodín v jednotlivých týždňoch a priemerný počet faktúrovateľných hodín na jedného zamestnanca vo výrobe, rozdelené podľa jednotlivých týždňov v roku,
- vývoj podielu celkového počtu faktúrovateľných hodín a celkových nefaktúrovateľných hodín v spoločnosti, rozdelené podľa jednotlivých týždňov v roku,
- filter pre zobrazenie dát pre konkrétneho zamestnanca, ktorý ovláda prvé dva grafy v reporte,
- filter pre zmenu prednastaveného reportovaného časového intervalu.

Nasledujúci obrázok zobrazuje stránku s reportom o výkone výroby spoločnosti.

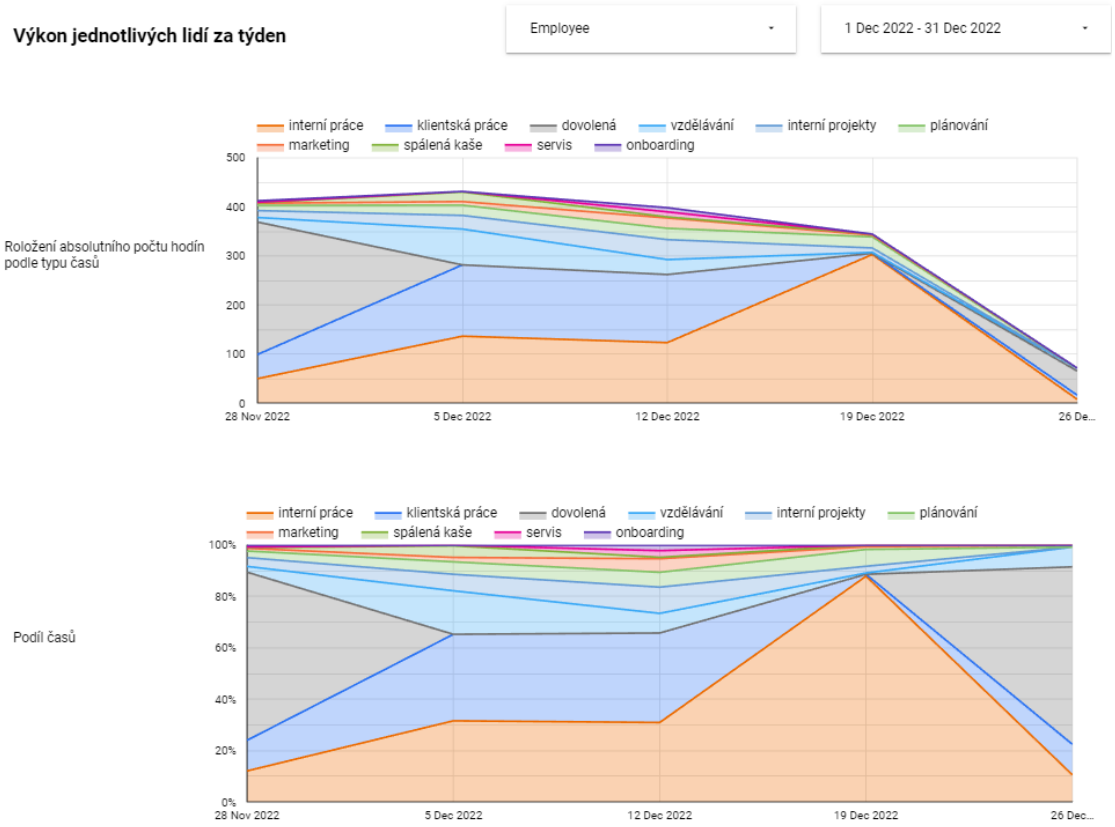



**Obrázok č. 56:** Report o výkone výroby (Vlastné spracovanie)

**Report o vykázaných časoch** zobrazuje celkové časy zamestnancami spoločnosti a ich rozdelenie podľa typov časov. Report obsahuje:

- vývoj celkového počtu vykázaných časov v spoločnosti za jednotlivé týždne v roku, rozdelené podľa štítkov (typov) časov,
- Relatívne zastúpenie typov časov na celkovom počte vykázaných časov, rozdelené na jednotlivé týždne v mesiaci,
- filter pre ovládanie a zobrazenie dát pre konkrétneho zamestnanca,
- filter pre zmenu prednastaveného reportovaného časového intervalu.

Nasledujúci obrázok zobrazuje stránku s reportom o vykázaných časoch spoločnosti.



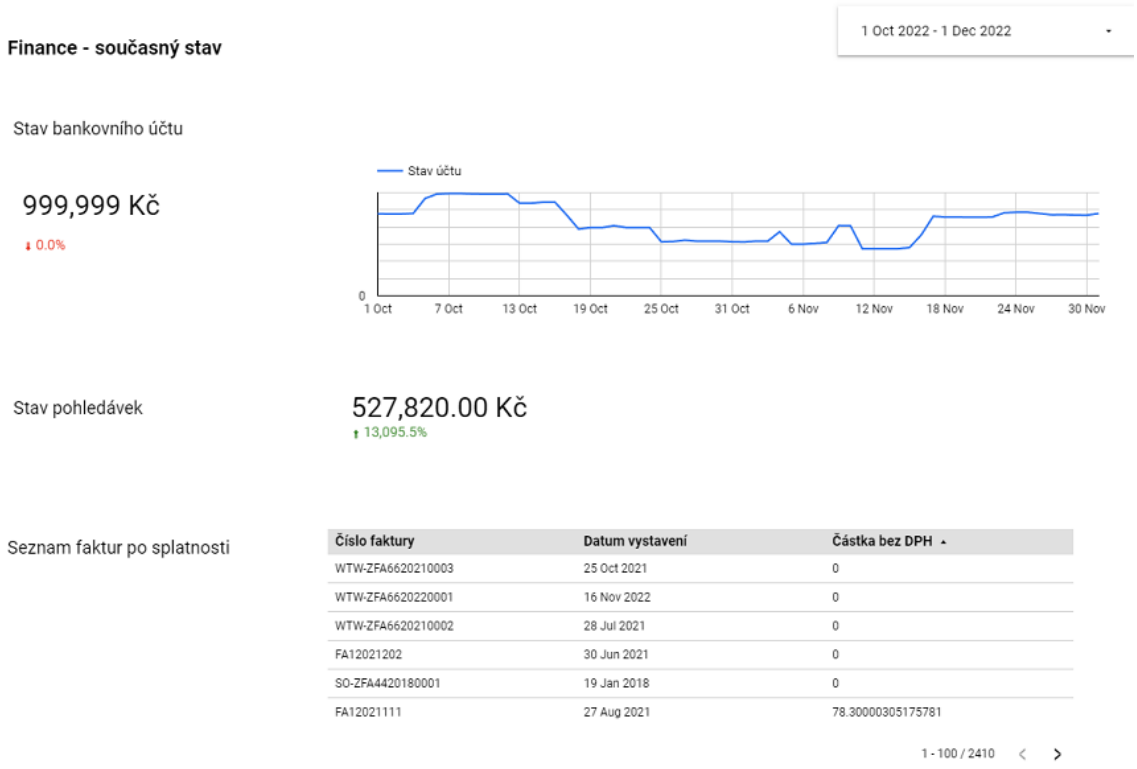

**Obrázok č. 57:** Report o vykázaných časoch (Vlastné spracovanie)

**Report financií** združuje a reporty informácie o aktuálnej finančnej situácií spoločnosti. Obsahuje:

- aktuálny (posledný známy) stav dostupných finančných prostriedkov na bankovom účte a ich vývoj v čase,
- stav všetkých existujúcich pohľadávok spoločnosti (vydaných nezaplatených faktúr),

- zoznam vystavených a neuhradených faktúr po splatnosti.

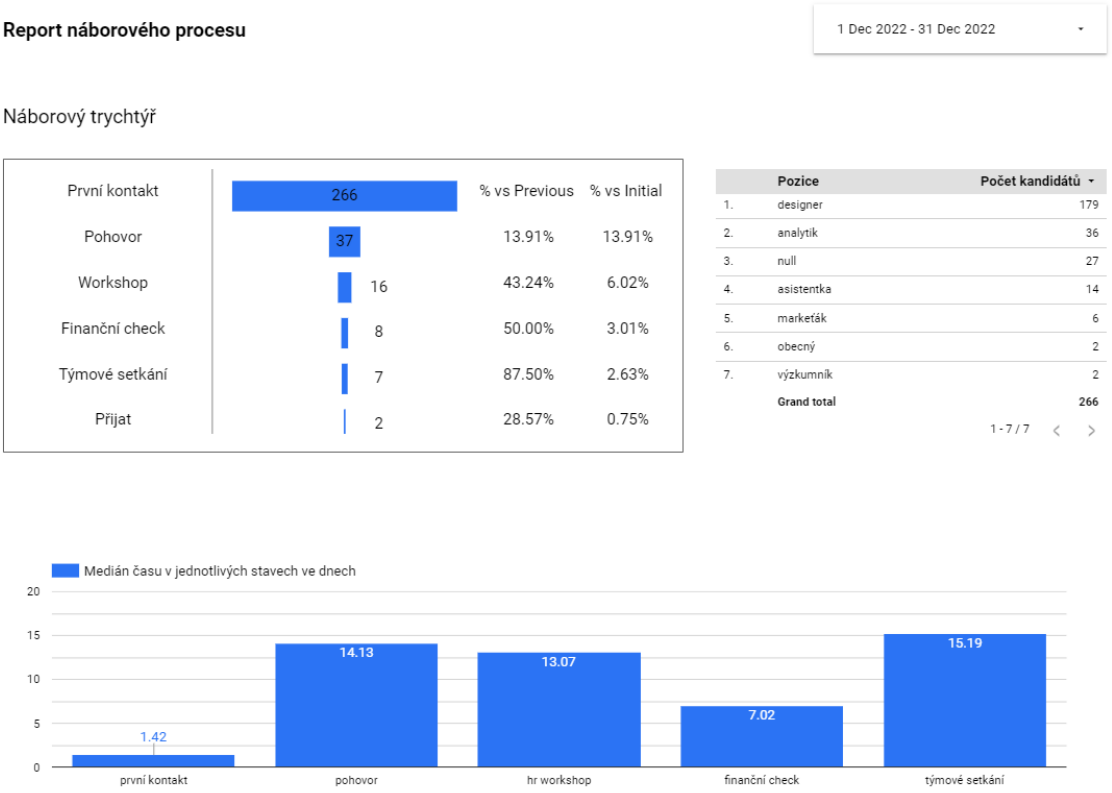
Nasledujúci obrázok zobrazuje stránku s reportom financií spoločnosti.




**Obrázok č. 58:** Report financií (Vlastné spracovanie)

**Report náborového procesu** je posledný implementovaný report. Prezentuje informácie o aktuálnom stave náborového procesu. Report obsahuje:

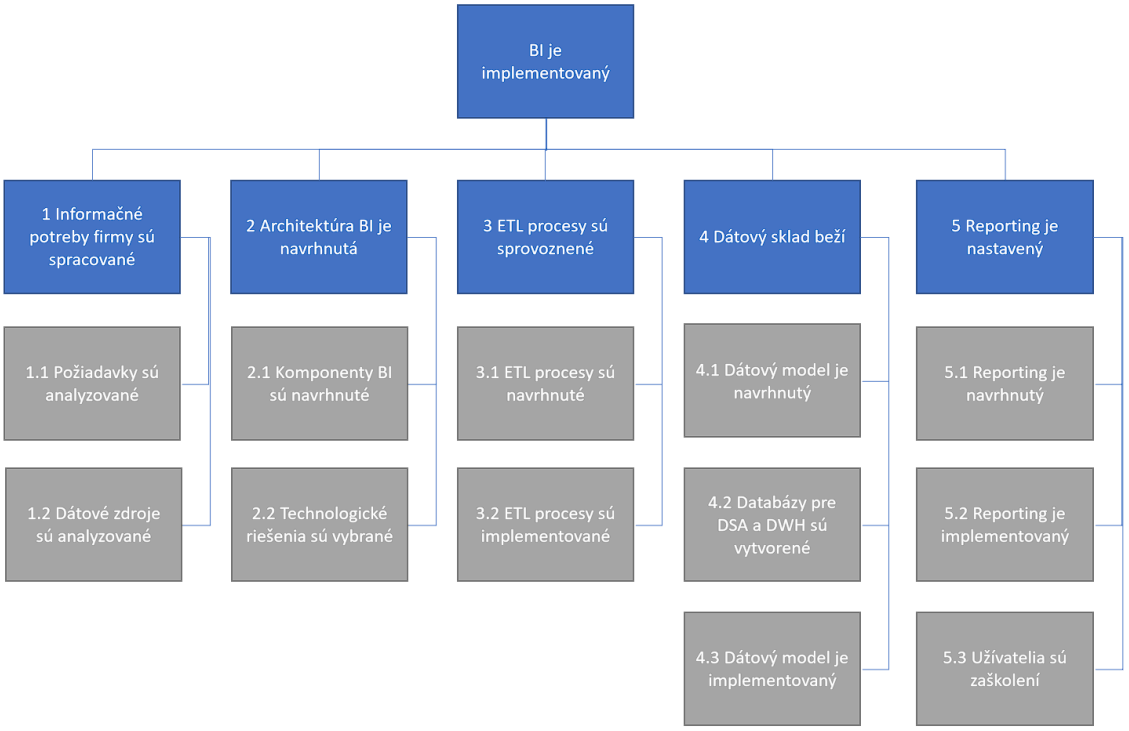
- celkový počet kandidátov na jednotlivé ponúkané pracovné pozície,
- vizualizovaný náborový lievik podľa jednotlivých stavov v náborovom procese pomocou lievikovej vizualizácie,
- zobrazený medián časov kandidátov v jednotlivých stavoch náborového procesu,
- filter pre zmenu prednastaveného reportovaného časového intervalu.




**Obrázok č. 59:** Report náborového procesu (Vlastné spracovanie)

## 3.5 Časové ohodnotenie návrhu a implementácie BI riešenia

Pre vypracovanie BI projektu a prezentácie časovej náročnosti implementácie BI riešenia je potrebné poznať odhadované časové náročnosti jednotlivých aktivít. Pre odhadnutie časovej náročnosti projektu je cieľ projektu rozložený na jednotlivé výstupy. Pre tento účel je použitý nástroj WBS (Work breakdown structure).




**Obrázok č. 60:** WBS BI projektu (Vlastné spracovanie)

Nasledujúca tabuľka obsahuje odhadované časové náročnosti jednotlivých výstupov podľa WBS v jednotke MD (manday, človekodeň).

**Tabuľka č. 4:** Časová náročnosť BI projektu (Vlastné spracovanie)

| Výstup | Časová náročnosť [MD] |
|---|---|
| 1.1 Požiadavky sú analyzované | 1 |
| 1.2 Dátové zdroje sú analyzované | 3 |
| 2.1 Komponenty BI sú navrhnuté | 2 |
| 2.2 Technologické riešenia sú vybrané | 2 |
| 3.1 ETL procesy sú navrhnuté | 2 |
| 3.2 ETL procesy sú implementované | 8 |
| 4.1 Dátový model je navrhnutý | 1 |

| | |
|---|---|
| 4.2 Databázy pre DSA a DWH sú vytvorené | 1 |
| 4.3 Dátový model je implementovaný | 1 |
| 5.1 Reporting je navrhnutý | 2 |
| 5.1 Reporting je implementovaný | 4 |
| 5.2 Užívatelia sú zaškolení | 1 |
| **Spolu** | **28** |

Odhadovaná časová náročnosť BI projektu je 28 MD (224 človekohodín).

## 3.6 Prínosy riešenia pre spoločnosť

Na základe realizovaných výstupov je možné konštatovať, že ciele práce, informačné potreby a požiadavky spoločnosti boli naplnené. Cieľový výstup riešenia BI, reporting, poskytuje automatizované a aktuálne informácie o výkonnosti firmy na jednom mieste. Reporting je prístupný a zrozumiteľný užívateľom, primárne vedeniu spoločnosti. V aktuálnom nastavení sú informácie poskytované reportingom oneskorené jeden deň. Vybrané technologické riešenia rešpektujú aktuálne používané technológie v spoločnosti. Celopodnikový dátový sklad poskytuje jednotné miesto pravdy, ktorý integruje každodenne generované podnikové dáta. Navrhnutý dátový model je štandardný a prispôsobený potrebám spoločnosti. V prípade migrácie do iného zdrojového systému je možné ETL procesy pretransformovať na nový systém a integrovať tak dáta na rovnaké miesto. Dátový sklad a uložené dáta v nich sú pripravené na prípadné budúce data mining algoritmy.

Po implementácií riešenia a 2 mesiacoch práce so systémom prebehol zber spätnej väzby od užívateľov riešenia BI. V nasledujúcej časti podkapitoly sa nachádzajú odpovede hlavného užívateľa riešenia BI, CFO, hlavného analytika a člena vedenia spoločnosti.

1. **Aké výhody prinieslo riešenie BI - dátový sklad a reporting?**

*"Obecně - aby lidé byli ochotni sledovat nějaká data, musí to pro ně být jednoduché a přínosné. Řešení nám přesně tohle dodal - během porad vedení jsme schopni během 10 minut zkontrolovat zdraví firmy a bavit se o výhledech pár měsíců dopředu. Tedy vlastně kontrolovat a řídit firmu."*

2. **Prinieslo riešenie nejaké nevýhody?**

*"Rozbil nám blažený pocit nevědomosti. A byl to poměrně velký projekt, který si vyžádal svůj prostor a investice. A nebyly úplně malé. Koncepce, implementace, návrh reportů. Bylo potřeba se s tím nějak naučit. Navíc se ukázalo, že první návrh vizualizací apod. nemusí být ten nejlepší, je třeba nad tím nějak iterovat a věci upravovat."*

3. **Uľahčilo vám prácu implementované riešenie? Ako veľmi? Ak nie, v čom bol problém?**

*"V první fázi, kde nyní jsme, je cílem projektu mít možnost během 15 minut zkontrolovat zdraví celé firmy. Marketing, obchod, výrobu, zaměstnance. Toho jsme v našem projektovém nástroji nebyli schopni. Tvorba reportů probíhala ručně, data byla na více místech, bylo třeba je dávat dohromady a čistit. To zabíralo práci nejexponovanějším lidem ve firmě. Proto se reporty moc nedělali a firmu jsme řešili víc na základě pocitů a náhodných impulzů."*

4. **Pomohlo vám BI riešenie v rozhodovaní? Ak áno, ako?**

*"Rozhodně, vidím tady velký posun v oblasti kontroly nastaveného systému. Vidět reporty o zdraví firmy nám umožňuje se o nich bavit - a hodnotit, jestli jsou věci v pořádku. Což ostatně pocítili i lidé ve firmě - začalo jim chodit spousta zvídavých otázek ve smyslu "proč projekt vychází tak, jak vychází" apod."*

5. **Ako často využívate reporty?**

*"Týdně."*

6. **Priestor na poznámky k implementovanému riešeniu.**

*"V další fázi bychom chtěli využít toho, že jsou data propojená. V rámci obchodu by se nám hodila data o tom, jaké sekvence kroků vedou k tomu, že firma nakonec poptá naše služby. Uvažujeme taky o dopočítání reálného zisku realizovaného na projektech či konkrétních zaměstnanců."*

# Záver

Cieľom tejto diplomovej práce bolo navrhnúť a implementovať riešenie Business Intelligence v malom podniku. Výstupom je kompletné a komplexné riešenie Business Intelligence, od integrovania dát z podnikových systémov pomocou ETL procesov do celopodnikového dátového skladu po reálne reporty podnikových metrík a výkonnosti.

Po zoznámení s oblasťou Business Intelligence a spôsobmi implementácie cez konkrétne technologické riešenia som sa zameral na informačné potreby podniku. Z analýz vyplynulo, že podnik nemalo podchytené reportovanie podnikových metrík a existujú konkrétne potreby a požiadavky pre efektívne riadenie firmy. Súčasťou tejto analýzy bola komplexná analýza zdrojových a ostatných systémov firmy.

Na základe týchto analýz som navrhol architektúru BI v malom podniku, vyvinul a implementoval jednotlivé komponenty BI. Konkrétne technologické riešenia komponentov boli vybrané na základe konzultácií so zodpovednými zamestnancami firmy. Boli vybrané technologické riešenia, ktoré sú vo firme dlhodobo používané a zamestnanci budú schopní ich samostatne rozvíjať.

Praktická časť práce obsahuje logické návrhy komponent BI - ETL procesov, dátového modelu dátového skladu, a technické implementácie komponent v príslušných technológiách. ETL procesy sú implementované pomocou vlastných skriptov kvôli komplexnému spôsobu uloženia dát v zdrojových systémoch, potrebe deduplikácie dát a neexistencie sofistikovaného a dostačujúceho riešenia na trhu ETL nástrojov. Implementácia ETL procesov takouto formou síce stala hodne investovaného času, ale prináša svoje výhody. Súćasťou implementácie BI je 5 reportov, ktoré odpovedajú potrebám na reporting v oblastiach výroby, financií, obchodného a HR procesov.

Hlavným prínosom týchto reportov a celkového BI riešenia pre spoločnosť je možnosť rýchlej, efektívnej, automatizovanej a pravidelnej kontroly zdravia a výkonnosti firmy. Riešenie BI otvorilo možnosť vedeniu riadiť firmu na základe dát, nastaviť procesy a systémy a kontrolovať ich. V budúcnu bude možné dátový sklad využiť na dolovanie znalostí pomocou sofistikovaných algoritmov.

Praktickú časť práce ukončujú vypracované WBS (Work Breakdown Structure), s časovým ohodnotením celého BI projektu, vlastné hodnotenie praktických prínosov BI riešenia firme a kladná spätná väzba od vedenia firmy o prínosoch riešenia.

# Zoznam použitej literatúry


[1] GROSSMANN, W. and S. RINDERLE-MA. Fundamentals of Business Intelligence. 1st ed. Heidelberg: Springer Berlin / Heidelberg, 2015. 366 p. ISBN 978-3-662-46530-1.

[2] LACKO, L. Databáze: datové sklady, OLAP a dolování dat s příklady v Microsoft SQL Serveru a Oracle. 1. vyd. Brno: Computer Press, 2003. 486 s. ISBN 80-7226-969-0.

[3] NOVOTNÝ, O., J. POUR a D. SLÁNSKÝ. Business Intelligence: jak využít bohatství ve vašich datech. 1. vyd. Praha: Grada, 2005. 256 s. ISBN 80-247-1094-3.

[4] SEIGE, V. Business intelligence: příručka manažera. 1. vyd. Praha: Tate International, 2007. 166 s. ISBN 978-808-6813-127.

[5] DAVENPORT, Thomas a Laurence PRUSAK. Knowledge: How Organizations Manage What They Know. II. Boston: Harvard Business Press, 2000. ISBN 0-87584-655-6.

[6] Treemap - Learn about this chart and tools to create it. The Data Visualisation Catalogue [online]. Copyright ©The Data Visualisation Catalogue [cit. 14.05.2023]. Dostupné z: https://datavizcatalogue.com/methods/treemap.html

[7] What does a box plot tell you? [online]. Manchester: McLeod, 2019 [cit. 2021-03-02]. Dostupné z: https://www.simplypsychology.org/boxplots.html

[8] PERMENTER, David. Key Performance Indicators: Developing, Implementing, and Using Winning KPIs. 4. Edition. Hoboken: John Wiley & Sons, 2019. ISBN 978-1119620778.

[9] ŠEDIVÁ, Zuzana. Vizualizace dat v návrhu dashboardu v oblasti BI. Systémová integrace. 2017, 2017(3), 52-62. ISSN 1804-2716.

[10] What is REST API | PHPenthusiast. Learn object oriented PHP | PHPenthusiast [online]. Copyright © 2015 [cit. 14.05.2023]. Dostupné z: https://phpenthusiast.com/blog/what-is-rest-api

[11] Our Documentation | Python.org. Welcome to Python.org [online]. Copyright ©2001 [cit. 14.05.2023]. Dostupné z: https://www.python.org/doc/

[12] SQLAlchemy - The Database Toolkit for Python [online]. Copyright © by SQLAlchemy authors and contributors. [cit. 14.05.2023]. Dostupné z: https://www.sqlalchemy.org/

[13] Welcome to Alembic's documentation! — Alembic 1.10.4 documentation. 302 Found [online]. Dostupné z: https://alembic.sqlalchemy.org/en/latest/

[14] Koch, Miloš. Datové a funkční modelování / Miloš Koch, Bernard Neuwirth. 4. rozš. vyd.. Brno : Akademické nakladatelství CERM, 2010. 142 s. brož. ( Učební texty vysokých škol [CERM] ) ISBN:978-80-214-4125-5

[15] Relační databáze vs. nerelační databáze: Čím se liší? | MasterDC. MasterDC – Specialisté na firemní IT infrastrukturu [online]. Copyright © 2023 MasterDC [cit. 14.05.2023]. Dostupné z: https://www.master.cz/blog/relacni-databaze-nerelacni-databaze-jake-jsou-rozdily/

[16] Looker Studio: Business Insights Visualizations | Google Cloud. Cloud Computing Services | Google Cloud [online]. Dostupné z: https://cloud.google.com/looker-studio

[17] ClickUp™ | One app to replace them all . ClickUp™ | One app to replace them all [online]. Copyright © [cit. 14.05.2023]. Dostupné z: https://clickup.com/

[18] GitHub - superfaktura/apiclient_cz: SuperFaktura API | Faktury online pro živnostníky a malé firmy. GitHub: Let's build from here · GitHub [online]. Copyright © 2023 GitHub, Inc. [cit. 14.05.2023]. Dostupné z: https://github.com/superfaktura/apiclient_cz

[19] TIWARI, S. Professional NoSQL. John Wiley Sons, 1. vydanie, September 2011. ISBN 978-0-470-94224-6.

# Zoznam použitých obrázkov

# Zoznam použitých tabuliek

# Zoznam použitých skratiek a symbolov

| | |
|---|---|
| BI | Business Intelligence |
| UX | User experience |
| UI | User interface |
| API | Application Programming Interface |
| ETL | Extract, Transform, Load |
| NAS | Network Attached Storage |
| CRM | Custom Relationship Management |
| ERP | Enterprise resource planning |
| CI | Custom Intelligence |
| EAI | Enterprise Application Integration |
| OLAP | On-line Analytical Processing |
| OLTP | On-line Transaction Processing |
| WWW | World Wide Web |
| SCM | Supply Chain management |
| DSA | Data Staging Areas |
| ODS | Operational Data Storage |
| DWH | Data warehouse |
| DMA | Data mart |
| SQL | Structured Query Language |
| KPI | Key Performance Indicator |
| HTTP | Hypertext Transfer Protocol |
| REST | Representational state transfer |
| XML | eXtensible Markup Language |
| JSON | JavaScript Object Notation |
| CSV | Comma Separated Values |
| ORM | Objektovo-relačné modelovanie |
| YAML | Yet Another Markup Language |
| HR | Human resources |
| CU | ClickUp |
| XLSX | Formát Microsoft Excel súborov |