SZAKDOLGOZAT

OE-NIK Hallgató neve: **Baji Botond**

2022 Hallgató törzskönyvi száma: **T-000123/FI12904**

Adatokadattárházba történő integrálása, majd azok webes felület segítségével történő vizualizálása

Integrating datas into DataWarehouse, visualize with web application

OE-NIK Hallgató neve: **Baji Botond**

2022 Hallgató törzskönyvi száma: **T-000123/FI12904**

Óbudai Egyetem
Neumann János Informatikai Kar
Kiberfizikai Rendszerek Intézet

SZAKDOLGOZAT FELADATLAP

Hallgató neve:	**Baji Botond**
Törzskönyvi száma:	T/005548/FI12904/N
Neptun kódja:	[redacted]

A dolgozat címe:

Adatok adattárházba történő integrálása, majd azok webes felület segítségével történő vizualizálása
Integrating datas into database, visualize with web application

Intézményi konzulens:	Légrádi Gábor
Külső konzulens:	
Beadási határidő:	2022. május 15.
A záróvizsga tárgyai:	Számítógép architektúrák Big Data és üzleti intelligencia specializáció

A feladat

A szakdolgozat kapcsán a hallgató feladata adattárházakba való adatok integrálása majd annak webes felület segítségével való vizualizálása.
Továbbá elérhetőnek kell lennie minden internetes felületen. Kutassa fel és vizsgálja meg milyen cross platform fejlesztésre alkalmas környezetek/nyelvek vannak jelenleg a piacon. Ismertesse a választott adatbáziskezelő rendszert és fejlesztői keretrendszert. Valósítson meg egy olyan alkalmazást, amely megfelel a megrendelővel közösen elkészítendő követelményspecifikációnak.

A dolgozatnak tartalmaznia kell:

- a feladat részletes ismertetését,
- a követelmény specifikációt,
- a rendszertervet, mely tartalmazza a funkciókat,
- a választott fejlesztői eszközök ismertetése és a választás indoklását,
- a továbbfejlesztési lehetőségeket

Ph.

.....
Dr. Fleiner Rita
intézetigazgató

A szakdolgozat elévülésének határideje: **2023. december 15.**
(OE TVSz 55.§ szerint)

A dolgozatot beadásra alkalmasnak tartom:

……..	…….
külső konzulens	intézményi konzulens

HALLGATÓI NYILATKOZAT

Alulírott hallgató kijelentem, hogy a szakdolgozat / diplomamunka saját munkám eredménye, a felhasznált szakirodalmat és eszközöket azonosíthatóan közöltem. Az elkészült szakdolgozatomban / diplomamunkámban található eredményeket az egyetem és a feladatot kiíró intézmény saját céljára térítés nélkül felhasználhatja.

Budapest, 2022.05.15

.....
hallgató aláírása

KONZULTÁCIÓS NAPLÓ

Hallgató neve: Neptun kód: Tagozat:

Baji Botond ..... [redacted] ..... Mérnök informatikus .....

Telefon: Levelezési cím (pl: lakcím):

[redacted] .....

Szakdolgozat / Diplomamunka[1] címe magyarul:

Adatok adattárházba történő integrálása, majd azok webes felület segítségével történő vizualizálása

.....

Szakdolgozat / Diplomamunka[2] címe angolul:

Integrating datas into DataWarehouse, visualize with web application

.....

Intézményi konzulens: Külső konzulens:

Légrádi Gábor ....

Kérjük, hogy az adatokat nyomtatott nagybetűkkel írja!

Alk.	Dátum	Tartalom	Aláírás
1.	**2020.03.20**	**A téma megbeszélése, az irodalomkutatás elkezdése**	Légrádi Gábor
2.	**2020.04.05**	**Követelmény specifikáció kialakítása**	Légrádi Gábor
3.	**2020.04.20**	**Rendszerterv létrehozása**	Légrádi Gábor
4.	**2020.05.10**	**Szakdolgozat végső formájának kialakítása**	Légrádi Gábor

A Konzultációs naplót összesen 4 alkalommal, az egyes konzultációk alkalmával kell láttamoztatni bármelyik konzulenssel.

A hallgató a Szakdolgozat I. / Szakdolgozat II. (BSc) vagy Diplomamunka 1 / Diplomamunka 2 / Diplomamunka 3 / Diplomamunka 4[3] tantárgy követelményét teljesítette, beszámolóra / védésre[4] bocsátható.

Budapest, 2021. 05. 16.

Légrádi Gábor
intézményi konzulens

KONZULTÁCIÓS NAPLÓ

Hallgató neve: Neptun kód: Tagozat:

Baji Botond ..... [redacted] ..... Mérnök informatikus .....

Telefon: Levelezési cím (pl: lakcím):

[redacted] .....

Szakdolgozat / Diplomamunka[1] címe magyarul:

Adatok adattárházba történő integrálása, majd azok webes felület segítségével történő vizualizálása

.....

Szakdolgozat / Diplomamunka[2] címe angolul:

Integrating datas into DataWarehouse, visualize with web application

.....

Intézményi konzulens: Külső konzulens:

Légrádi Gábor ....

Kérjük, hogy az adatokat nyomtatott nagybetűkkel írja!

Alk.	Dátum	Tartalom	Aláírás
1.	2021.10.20	Rendszerterv megvalósítása	Légrádi Gábor
2.	2021.11.05	Környezet kialakítása	Légrádi Gábor
3.	2021.12.01	Az elkészült rendszer dokumentálása	Légrádi Gábor
4.	2021.12.15	Szakdolgozat II. bemutatása	Légrádi Gábor

A Konzultációs naplót összesen 4 alkalommal, az egyes konzultációk alkalmával kell láttamoztatni bármelyik

[1] Megfelelő aláhúzandó!
[2] Megfelelő aláhúzandó!

konzulenssel.

A hallgató a Szakdolgozat I. / Szakdolgozat II. (BSc) vagy Diplomamunka 1 / Diplo-mamunka 2 / Diplomamunka 3 / Diplomamunka 4[3] tantárgy követelményét teljesítette, beszámolóra / védésre[4] bocsátható.

Budapest, 2021.12.15.....

Légrádi Gábor

.....

intézményi konzulens

[3] Megfelelő aláhúzandó!
[4] Megfelelő aláhúzandó!

Tartalomjegyzék

Táblázatjegyzék

Ábrajegyzék

1. Bevezetés

Szakdolgozat témája a 2019-ben megjelent COVID-19 (" korona vírus") vírusról való információ szolgáltatás. A cél egy olyan alkalmazás létrehozása, amelyben a vírusról valós idejű információkat gyűjt be. A COVID-19 alkalmazásban a halálesetek bemutatása, bejelentett gyógyulások, valamint a vírus elterjedését mutatja be [1].
Az adatok országok szerint kerülnek felbontásra és egy adott országon belül pedig régiókra.
Több mint 110 országon és területen belül kerülnek lekérdezésre az adatok.
Az adathalmazon világszerte követem a COVID-19 hatását az egyes országokon belül. Annak érdekében, hogy megértsük a járványt, meg kell határozni a karakterisztikáját és a viselkedését. Ezt az a rendelkezésre álló adatok begyűjtésével és analizálásával tesszük meg. Erre a Big Data eszközrendszere a legmegfelelőbb megoldás.

1.1 A COVID-19

A járvány világszerte rengeteg emberi áldozattal jár, jelentős gondot okoz a gazdaságban, a közösségi életben és az egészségügyben. A pandémia gyors terjedése világszerte megfertőzött több, mint 82 millió embert. A vírus terjedését a felépítésének ismeretének hiányában, valamint a gyors mutációknak következtében, egyre nehezebb kontrolálni [2].
A koronavírus fertőzést 2019 -ben a WHO világjárványként definiálta.
Először Kínában jelentették be a vírust, azon belül Wuhan városában. A COVID génláncait visszafejtették és sok egyezést találtak a 2003-ban kitört SARS járvány vírusával. Nemhiába a hasonlóság, így kapta a COVID-2 elnevezést.
Az eredete a SARS-CoV-2 vírusnak még mindig ismeretlen, de feltételezték, hogy a wuhani piacon árult kígyókról, madarakról, vagy denevérekről terjedhetett át az emberre. Végül a denevérekhez kötötték a vírust, ugyanis a denevérben is megtalálható vírus génállományában 96%-os egyezést találtak. A denevér kolóniát 2000 km-re Wuhantól azonosították. Az első hullám óta a fertőzés terjedése felgyorsult a világban. Az European Centre for Disease Prevention and Control szerint 2020.06.07 óta 8,142,129 fertőzést jelentettek be világszerte, amiből 443,488 a haláleset. Akkoriban Amerika volt az legjobban fertőzött kontinens.

1.2 Big Data impact on COVID-19

Big Data számos lehetőséget nyújt a vírus terjedésének, aktivitásának modellezésében. Ez segítséget nyújt az egyes országoknak, hogy időben felkészüljenek a járvány terjedésére.

the Official Aviation Guide	**the location-based services of the Tencent**	**Wuhan Municipal Transportation Management Bureau**

Az alábbi felsorolt adatbázisok hatékony segítséget nyújtanak a vírus terjedésének modellezéséhez [3]. Ezek adatokat tárolnak a halálesetekről, fertőzések számárólés a gyógyulásokról is. A Big Data alkalmazásával rengeteg információt lehet gyűjteni, elemezni, amely segíti a tudósok, egészségügyben dolgozók, epidemilógusok munkáját. A begyűjtött adatok segítségével elemezni tudják a vírus terjedését és segít az ellenszer kidolgozásában. Big Data analízis fontos szerepet játszik a vírus terjedésének követésében, a kutatásában és a megelőzésben is.

2. Követelmény specifikáció

Szakdolgozat során olyan alkalmazás kerül kifejlesztésre, amely Real-Time, azaz valós idejű adatokból nyer információt és azokból készít riportokat, modelleket, analízist, lekérdezéseket. Az adatok CSV fileokból kerülnek importálásra. Ezeket az adatok adatbázisba kerülnek. Megfelelő normalizálások után, kiválasztva a megfelelő Big Data architektúrát, az adatok adattárházba kerülnek be.

2.1 Adatbázis

Specifikáció alapján meghatározásra kerültek azok az adathalmazok, amelyeket az adatbázisban létrehozok. A tábla szerkezeteket EK diagram segítségével ábrázolom.

Ezáltal könnyebben átlátható adatstruktúrát kapunk. Az EK diagram használata után láthatjuk az adott táblákat és a közöttük lévő kapcsolatokat.

Az adatok CSV file-ban tárolódnak és PostgresSQL segítségével a megfelelő tábla szerkezetek létrehozása után, beimportáltam [4].

2.1.1 Táblák

A **countries_aggregated** tábla tartalmazza az egyes országokban a korona vírus jelenlétét. Az adathalmazban a dátum mező a későbbi feldolgozást segíti a riportok, analízis készítésében.

A tábla szerkezete:

Oszlop	Adat típus	magyarázat
date	dátum	A korona vírus hatása dátum szerint országokra bontva
country	Varchar(50)	Az ország neve
confirmed	int	Bejelentések száma
recovered	int	Felépültek száma
deaths	int	Halál esetek száma

1: Countries aggregated

A **key_countries_pivoted** tábla tartalmazza azokat az országokat, ahol a korona vírus jelenléte a legnagyobb volt az adott dátum szerint.

A tábla szerkezete:

Oszlop	Adat típus	magyarázat
date	date	A korona vírus hatása dátum szerint országokra bontva
China	int	Kínában a fertőzések száma
U.S	int	U.S-ban a fertőzések száma
United_kingdom	int	United_kingdom -ben a fertőzések száma
Italy	int	Olaszországban a fertőzések száma
France	int	Franciaországban a fertőzések száma
Germany	int	Németországban a fertőzések száma
Iran	int	Iránban a fertőzések száma

2.key_countries_pivoted

A **US_confirmed** tábla United_states-ről tartalmaz információkat. A tábla államokra bontva, azon belül települések szerint nyújt tájékoztatást a vírusról.

A tábla szerkezete:

Oszlop	Adat típus	magyarázat
country_name	Varchar(50)	Államokban az egyes települések nevei
date	date	Vírus dátum
country_region	Varchar(50)	Az ország régiója
cases	bigint	Ügyek száma
province_state	Varchar(50)	Államok neve

3.us_confirmed

A **US_simplified** táblát átalakítottam, ahol már a halálesetek száma is látható. Dátumhoz kötve tárolódnak az adatok és azokhoz kapcsolódnak az esetek.

A tábla szerkezete:

Oszlop	Adat típus	magyarázat
country_name	Varchar(50)	Államokban az egyes települések nevei
date	date	Vírus dátum
country_region	Varchar(50)	Az ország régiója
cases	bigint	Ügyek száma
province_state	Varchar(50)	Államok neve
deaths	bigint	Halál esetek száma

4.us_simplified

A **world_wide** nevű táblában az egész világon összesített adatok tárolódnak.
A fent említett tábla szerkezete:

Oszlop	Adat típus	magyarázat
date	date	Dátum a korona vírus helyzetéről napról napra
confirmed	bigint	Vírus fertőzöttek száma
recovered	bigint	Felépültek száma
deaths	bigint	Halál esetek száma
increase_rate	bigint	Növekedési ráta

5.worldwide

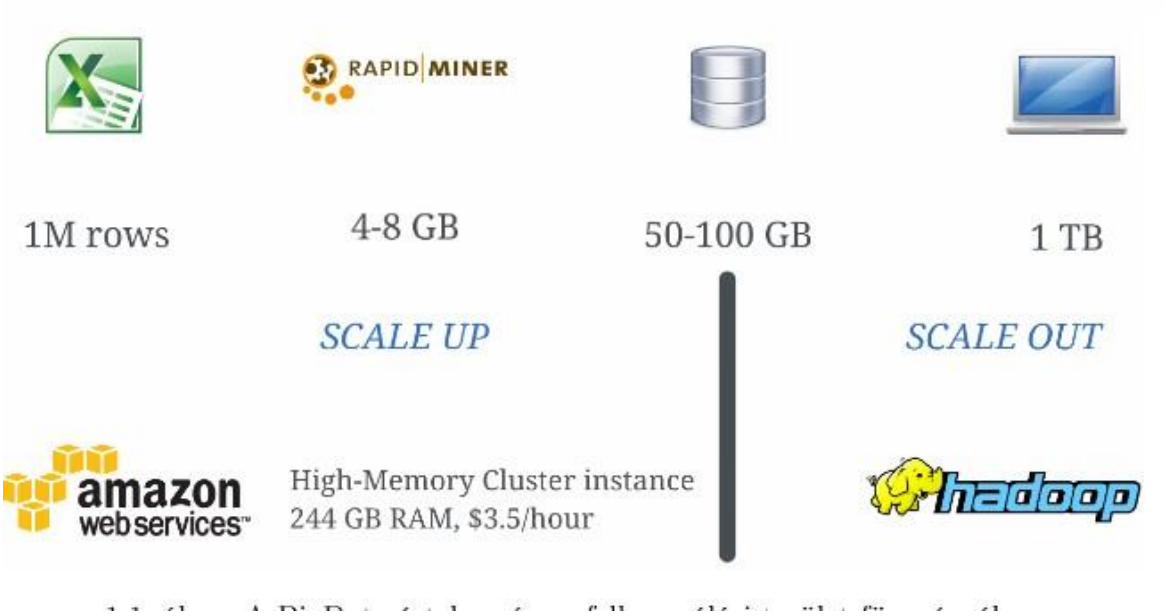
6.reference

Oszlop	Adat típus	magyarázat
uuid	int	Egyéni azonosító az összes rekordra
Iso2	Varchar(10)	2 betűs ország azonosító
Iso3	Varchar(10)	Hivatalos ország azonosító
Code3	int	
FIPS	bigint	Szövetségi információ feldolgozó kód, ami egyénileg azonosítja az országot az USA-n belül
Admin2	Varchar(50)	Ország név
Province_state	Varchar(50)	Állam neve
lat	Double precision	Földrajzi szélesség
long	Double precision	Földrajzi hosszúság
population	bigint	Népesség populáció
Combined_key	Varchar(50)	Államok országokra bontva
Country_region	Varchar(50)	Ország régió

3.Big Data architektúra

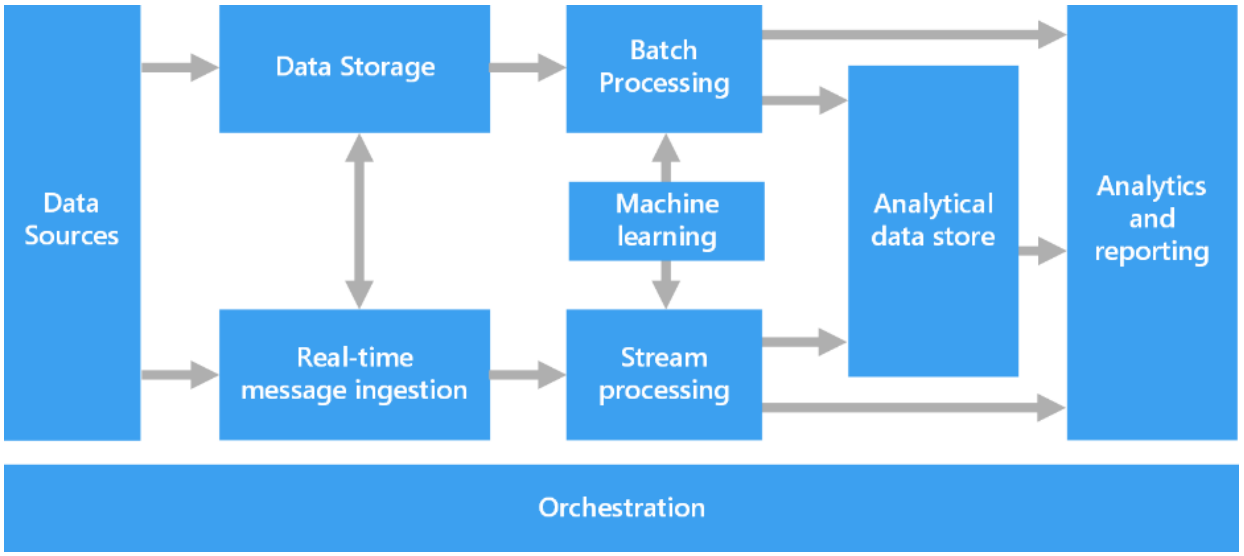
A Big Data típusú architektúrát olyan adatok betöltésére, feldolgozására tervezték, amelyek túl nagyok vagy összetettek lennének egy hagyományos adatbázis rendszer számára [5]. Az évek során megváltozott az adatkörnyezet. Az adatokat másképp kezelik

és az ezekkel kapcsolatos elvárások is mások lettek. Bizonyos adattípusok gyorsan gyűlnek. Előfordulnak összetett elemzési problémák is, vagy olyanok, amelyekhez nélkülözhetetlen a Big Data világa.



7.Big Data Components

3.1 A Big Data -architektúrák összetevői



8.Big Data Process

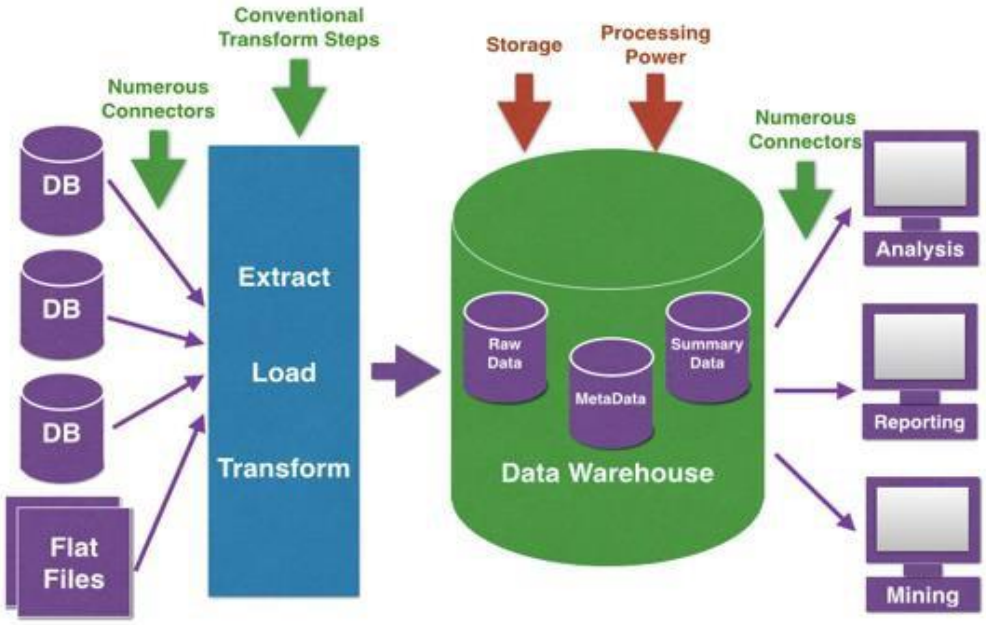
- **Adatforrások**: Minden Big Data megoldás egy vagy több adatforrás meghatározásával kezdődik. A szakdolgozatban használt PostgreSQL, amely relációs adatbázis.
- **Adattárolás:** A kötegelt feldolgozáshoz használatos adatokat egy elosztott fájltároló tartalmazza (ETL), amely nagy méretű adatokat képes tárolni a számítógépen.
- **Elosztott adattároló központok:** A szakdolgozat során az adatokat előkészített módon adom át, hogy lekérdezhetőek legyenek elemző eszközökkel. Ezeket a rendszereket kifejezetten nagy mennyiségű adatok tárolására fejlesztették ki.
- **Elemzés és jelentés készítés:** A legtöbb Big Data megoldás célja, hogy a lekérdezéssel, jelentésekkel jobban megérthetőek legyenek az adatok. Az architektúrában OLAP modellt fogok használni.

4. Adatmigráció

Az adatmigráció az a folyamat, ahol információt és adatokat gyűjtünk be.

A hagyományos adatmigráció az ETL (extract, transform, load) folyamatokon alapszanak. Ahhoz, hogy az adatokat integrálni tudjuk, kell egy forrás, ahonnan az adatokat kapjuk, és egy másik környezet, ahová exportáljuk az adatokat. ETL folyamatok három fontos funkcióval bírnak, hogy az adatokat megfelelően összegyűjthessük.

- **Extract**: Az adatokat beolvassa a forrás adatbázisból.
- **Transform**: átformálja az adatokat a cél adatbázisba, hogy minden adat az előírásoknak megfeleljen.
- **Load**: Az adatokat a cél adatbázisba írja.



9.Traditional Data Integration

5. Adatmodellezés

Az adatmodell: olyan fogalmak halmaza, amelyek az adatbázis struktúráját írják le. Az adatbázis adatmodelljeit 3 kategóriába soroljuk [7].

1. A koncepciós szintű adatmodellek a felhasználói szinten adatleírás.
2. Logikai szintű adatmodellek függenek az adatbázistól, de ezek még mindig absztraktak.
3. A fizikai szint adatmodelljei már teljesen függenek az adatbázistól, és azt írják le, hogyan tároljuk fizikailag az adatokat. A szakdolgozat során én ezt használom.

5.1 ROLAP architektúra

ROLAP architektúra esetén a modellünk a hagyományos relációs adatbázis környezetben szeretnénk megvalósítani, a kívánt adatbázis szerkezettel.

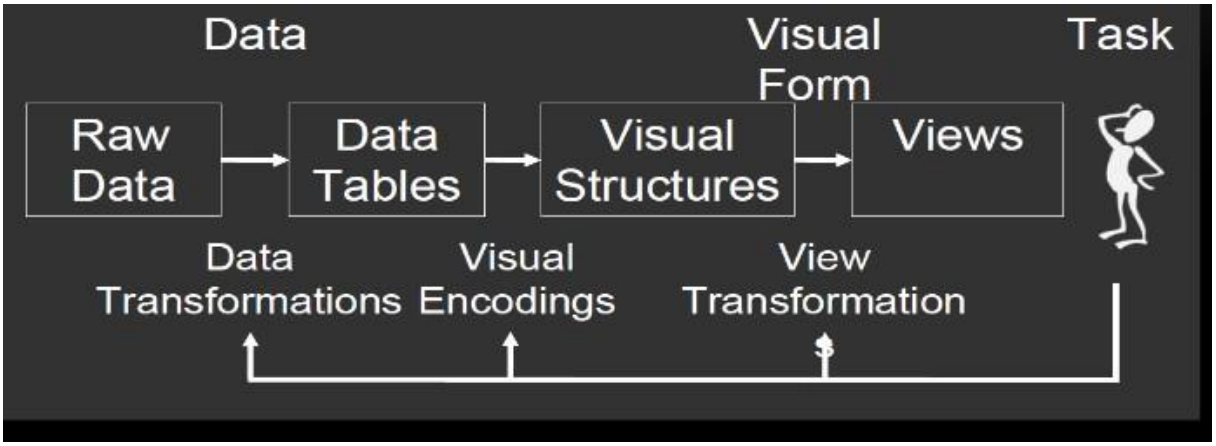
A most tárgyalt, Ralph Kimball által részletesen leírt módszerek mára kvázi „ipari szabványnak” számítanak [7].

5.2 Relációs adattárház séma tervezésének 4 lépéses folyamata

1. Modellezendő üzleti folyamat kiválasztása.
2. Felbontás meghatározása.
3. Dimenziók meghatározása.
4. Tényadatok meghatározása.

6. Adatvizualizáció

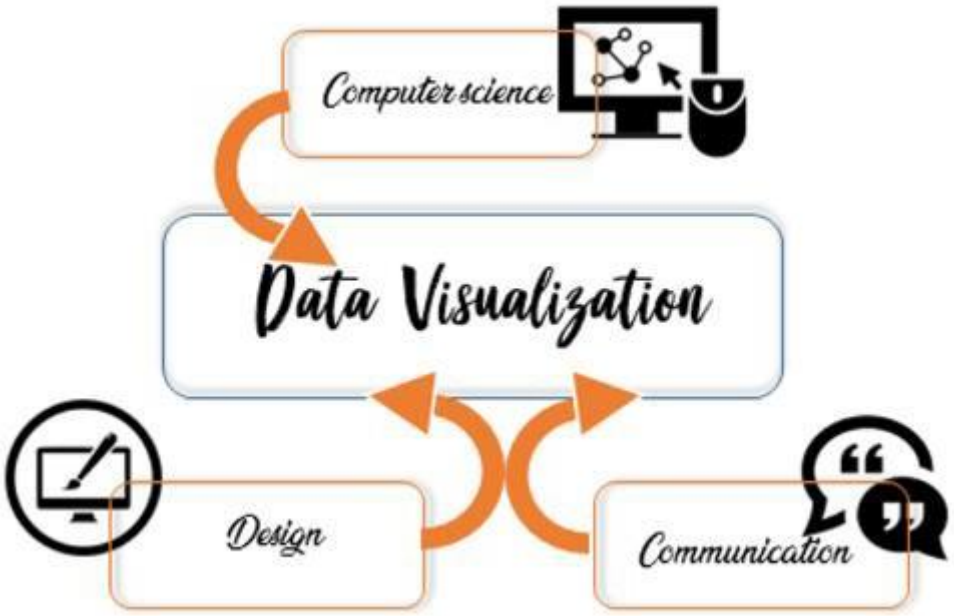
Adatvizualizáció az a folyamat, ahol grafikusan ábrázoljuk az adathalmazunkat a jobb megértés érdekében. Ez adatpontokat használ, mint például: gráfok, kör diagram, tartalom [8].



10.Data Vizualizations

6.1 Adatvizualizáció működése

A legtöbb adatvizualizációs szoftver képes csatlakozni adatforrásokhoz még relációs adatbázisban is. Ezek az adatok az adatbázis szerveren tárolódnak, a későbbi analízishez. A felhasználó szabadon megválaszthatja, a megjelenítés módját. Néhány szoftver automatikusan ajánlja a legjobb megoldásokat [9].



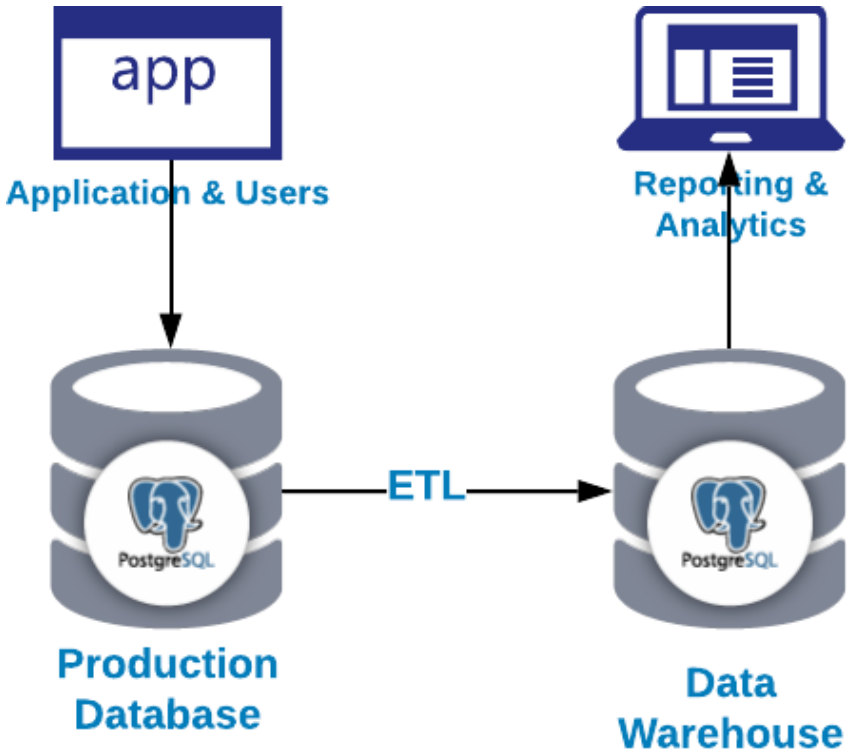
11.adatvizualizációs modell

7. A választott fejlesztői eszközök ismertetése és a választás indoklás

7.1 PostgreSql

PostgreSQL a legjobb funkciókkal rendelkező Open Source adatbázis. Flexibilis relációs adatbázisként tudom kezelni, alacsony költségű adattárház megoldásokat is nyújt. Számos előnye van PostgreSQL-t használni adattárházak esetén is[10].

- Költség: Szakdolgozat során adatbázis szerverem, ingyenesek.
- Skálázás: Replika node-ok használata végtelen, így jól skálázható.
- Teljesítmény: Megfelelő konfiguráció esetén, gyors teljesítményre képes.
- Kompatibilitás: Integrálhatóak az adatok, adatbányászathoz, OLAP, és riport készítéshez is.
- Kiterjeszthetőség: PostgreSQL-nek felhasználó barát adattípusai és függvényei vannak.



12.PostgreSQL with Datawarehouse

7.1.1 Konfiguráció beállítása adattárház használathoz

Nincs minden esetben optimális konfiguráció az adatbázishoz. Számos faktoron múlik, de pont emiatt választottam ezt az adatbázis kezelő rendszert [10].

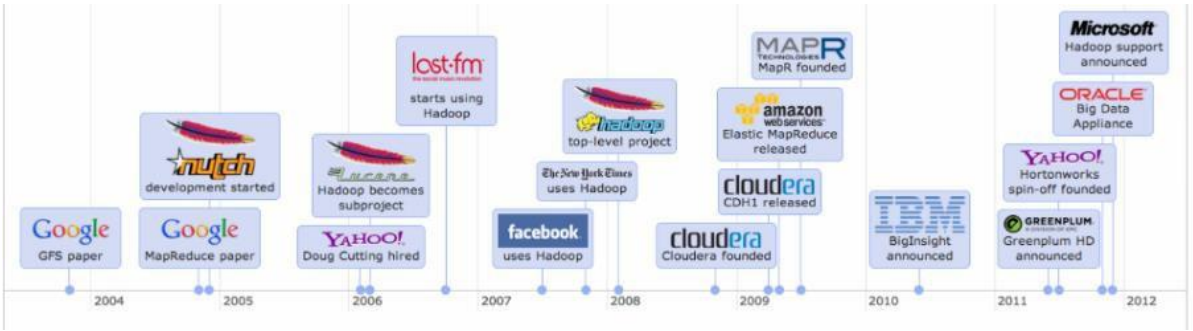
Memória alapú	CPU alapú
Maximum kapcsolódás: Nincs szükség nagy számú adatbázis kapcsolatra, csak riportok és analízisek készülnek	**Maximum processzor**: Be állítható a háttérben dolgozó processzorok száma.
Memória felszabadítás: Beállítható, hogy annyi memóriát használjon a gép amennyit a szerver használ fel.	**Maximum párhuzamosítható szálak**: Átalában ezt a beállítást annyira kell állítani, amennyit a számitógépünk támogatni tud.

Konfiguráció

8. Hadoop

8.1 Hadoop története

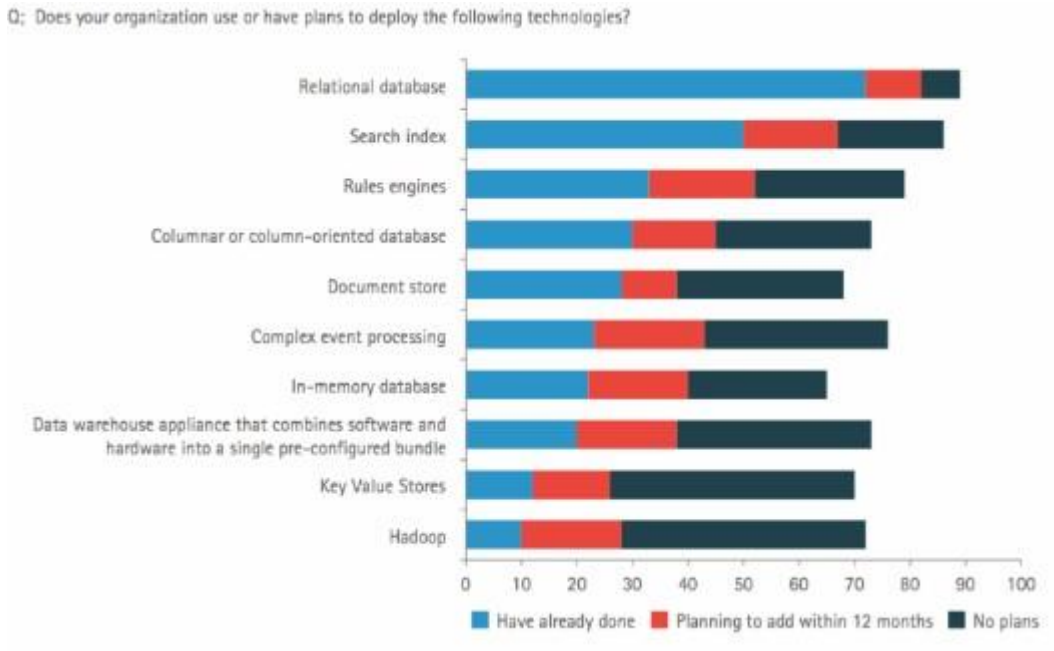
Eredetileg a Hadoop egy elosztott fájlrendszerként indult, amelyet a Google kezdett el fejleszteni Google File System néven. Ezt többen megpróbálták reprodukálni Open Source projektként *[6]*.



13.Hadoop kialakulásának története

Ezek után a Yahoo is elkezdte ugyanezt a rendszert elkészíteni.

A fejlesztések után egyre többen kezdték el használni a Hadoop rendszert. Ennek köszönhetően az Apache felkarolta, és egy dedikált Open Source fejlesztési projekt keretein belül használja mai napig. A 6.1 ábrán látható, hogy a rendszert céges környezetben nagyon sokan használják. A szakdolgozatom során én is a Hadoop renszereit használom [6].



14.Hadoop felhasználási terület

8.2 Hadoop HDFS

A Hadoop rendszer egy teljesen saját, elosztott fájlrendszert használ,Hadoop Distributed File System néven. Ez a fájlrendszer kifejezetten arra lett kitalálva, hogynagyméretű fájlokat tudjunk rajta tárolni, lekérdezni, vagy műveleteket végre hajtani rajtuk. Arra is figyeltek, hogy átlagos otthoni PC-kre is telepíthető legyen. Ezzel költség hatékonyan lehet elosztott rendszert kialakítani. A Szakdolgozatsorán ez tökéletesen megfelel a követelményeknek. A HDFS egy blokkalapú fájlrendszer tipikusan 64-256 MB egy blokk méret, tehát a nagy fájlok tipikusan GB nagyság rendűek lehetnek[6].

8.3 Hadoop csomagok

Minden csomag vagy projekt Open Source, az Apache oldalán megtalálhatóak. Szakdolgozatomban ezeket fogom használni.

15.Hadoop csomagok

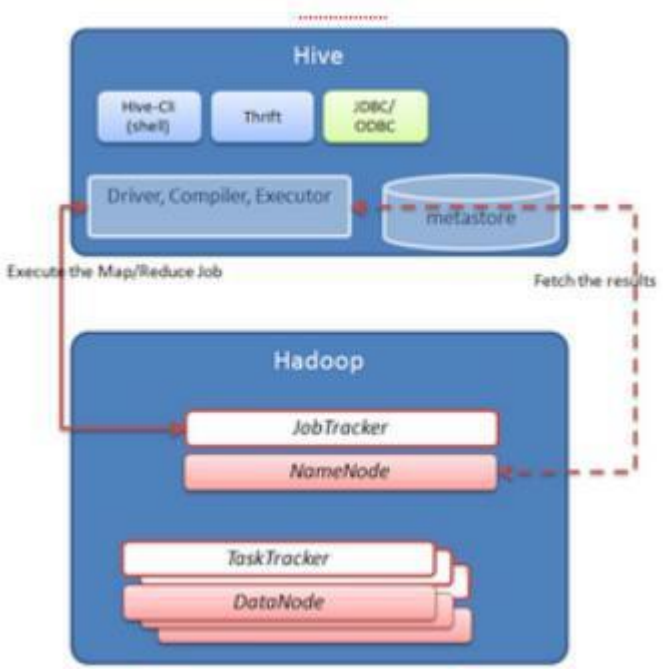
8.3.1 Sqoop

Különböző SQL alapú adatbázisokkal képes adatot cserélni. Az adatbázis PostgreSQL, amely megfelel ehhez a feladathoz. A Sqoop JDBC kapcsolaton keresztül éri el a Postgres adatbázist, és képes akár 2-3 gépből álló PostgreSQL clustert kiszolgálni. A Sqoop ezen a táblákat átmásolja a HDFS-re, majd elindít egy MapReduce folyamatot, így képes adatot másolni az adatbázisból HDFS-re. Ez természtesen két irányban is működik, de erre nem lesz szükség a szakdolgozat elkészítése során. A folyamat egyszerűen az, hogy elindul egy Mapreduce job, a JDBC connection kiépül és az adatok lehívásra kerülnek.

8.3.2 Hive

Ez a csomag már az elemzési réteghez tartozik, míg az előbbiek az adatelérési réteghez tartoztak. A Hive egy SQL lekérdező nyelvet nyújt a Hadoop felé.

Hivenak az a feladata, hogy SQL lekérdezéseket hajtson végre. Az SQL lekérdezésekből csinál egy MapReduce job-ot és ezt lefuttatja a clusteren, majd visszaadja az eredményeket. Ez megkönnyíti azok, munkáját, akik jártasak az SQL nyelvben. Ezt tipikusan az analitikus lekérdezésekhez használják. Szakdolgozat során ez nagyban hasonlít a PostgreSQL lekérdezéseihez [6].



16.Hive működése

6. Qlik Sense

A Qlik Sense egy gyors, rugalmas interaktív elemzési lehetőséget biztosító rendszer, amellyel több száz felhasználó nagy adatmennyiséget is egyidejűleg képes feldolgozni. Mondhatnánk, hogy egy egyszerű BI eszköz, de azért annál jóval több [11].

Qlik Sense segítségével számos adatvizualizációs eszköz elérhető a szakdolgozatom során. Könnyen skálázható, az adatbetöltés Hadoop segítségével könnyen megvalósítható. Az alábbi táblázat demonstrálja miért ezt választottam.

Capability	Qlik Sense	Power BI	Tableau
Free Form Exploration	●	◑	◑
AI-Powered Recommendations	●	◑	◕
Fully Functional Mobile Experience	●	◑	◕
Scalability	●	◑	◕
Embedding Visualizations	●	◑	◕
Deployment Flexibility	●	◑	◑
Governed Self-Service	●	◕	◑
Single Interface for Broad Use Cases	●	◑	◑
Data Integration	●	◑	◑
Supporting Data Literacy	●	◕	◕
Total Cost of Ownership	●	◑	◕

17.Összehasonlítás más adatvizualizációs eszközökkel

10.Rendszterv megvalósítása

A következő fejezetben a rendszer kialakításáról, tervezéséről, a környezetről, a feladat implenetációjáról, program működéséről lesz szó.

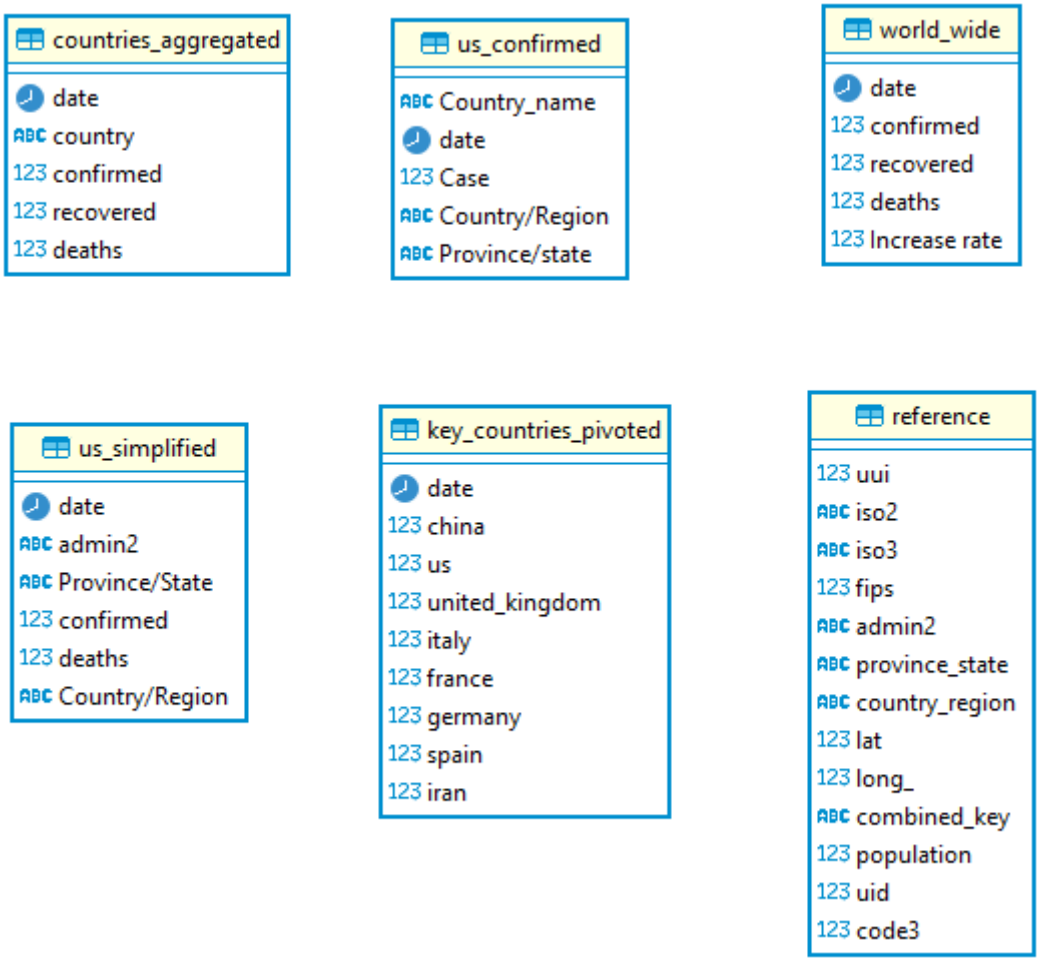
10.1 Környezet kialakítása

Első lépésként az adatbázist készítettem el. Az adatbázis DBeaver (21.0.4) asztali alkalmazásban, azon belül PostgreSQL szerveren fut. Adattárház megoldáshoz Hadoop-ot választottam, ugyanis, ha a későbbiekben változtatni szeretnék a programon könnyen átskálázható. Hadoop-ot Windows 10 alatt telepítettem. Java 1.8 JDK szükséges volt hozzá, mivel a Hadoop Java alatt fut.

Az adatokat migrálásához relációs adatbázisból az adattárházamba, szügség volt egy ETL eszközre, amely jelen esetben az SSIS. Az adatok streamelését Kafka segítségével oldottam meg ami egy Hadoop komponens. A Hive (2.0) az adatok lekérdezéséhez kellett, amely az adattárházhoz kapcsolódik. A riportok és az adatok elemzéséhez a Qlik Sens Desktopot használtam, amivel könnyedén tudtam csatlakozni az adattárházamban lévő adatokhoz.

10.2 Adatbázis megvalósítása

Az adatbázis specifikáció alapján meghatározhatóak voltak azok az adathalmazok, amelyek az adatbázishoz elengedetlenek voltak. Első lépésként magát az adatbázist készítettem el PostgreSQL segítségével. Az alkalmazás Localhoston fut, de a munkám során szükségem engedélyeznem tcp/ip cím kapcsolatot is, ugyanis a Hadoop-ot csak így értem el. Az adatbázis sémám a Covid19 nevet kapta. Az adatbázist létre hozó scriptek a mellékletekben látszódnak. A táblák a [10]. ábrán láthatóak.



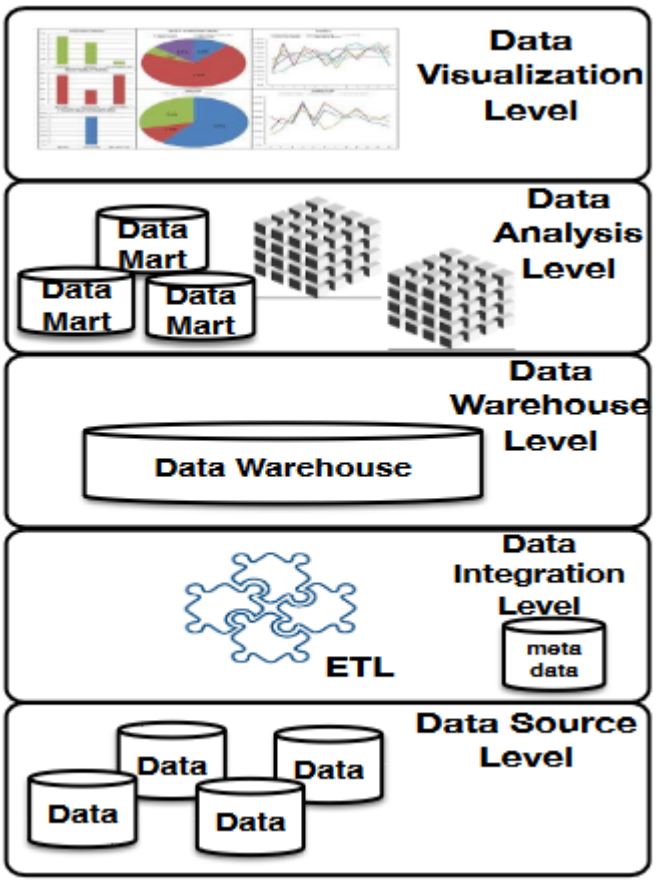
18.Tábla szerkezet

10.3 Covid adattárház: design

Az adattárházak fő feladata az adatintegritás és a konzisztencia, ugyanis az adatok heterogén adóforrásokból érkeznek. A hagyományos adattárházak úgy kerültek kialakításra, hogy nagy mennyiségű hisztorizált adatot lehessen rajta elemezni. Ennek létrehozása a következő lépésekből áll: adatfeldolgozás adatbázisból, vagy adatbázisba való adatbetöltés. Előfordul, hogy az adatok nincsenek normalizálva, akkor 3. normálformába kell hozni őket. Ez a metodológia segít csökkenteni az adat duplikációt, biztosítja az adatok közti integritást. Adat transzformáció elengedhetetlen, segít az adatokat denormalizálni, és az adattárházba már az így átalakított adatok kerülnek be. Adat betöltésénél csillag séma modell szerint jártam el. A csillag séma centralizál, létrehoz egy tény táblát ahová számos dimenzionális tábla kerül kerül összekapcsolásra.

Az adattárház architektúrám 5 rétegből áll össze:

a. **Adat forrás**
b. **Integrációs szint**
c. **Adattárház szint**
d. **Analízis szint**
e. **Adatvizualizációs szint**



19.Adattárház architektúra

10.4 Covid adattárház: implementáció

A Hadoop környezetem Windows 10 operációs rendszer alatt futtatom.
A Hadoop telepítése elég komplikált Windows 10 alatt ezért lépésről lépésre dokumentáltam a folyamatot.
A szükséges eszközök először is a JAVA JDK volt ugyanis a Hadoop Java környezet alatt fut.
CMD vagy PowerShell kellett, hogy teszteljük a környezeti változókat és futtassuk a Hadoop-ot.
A Java 1.8-as verziója már telepítve volt, így azzal nem volt gondom.

1. Környezeti változók beállítása:
   Először a JAVA_HOME nevű környezeti változót állítottam be, ahol az alap könyvtár a következő volt: **C:\Program Files\Java\jdk1.8.0_261**. Ugyanezeket a beállításokat elvégeztem a HADOOP_HOME változóra is.

2. Hadoop konfigurálása:
   A legfontosabb részeket kell bekonfigurálni, amik a következők:
   - Core
   - Yarn
   - MapReduce
   - Hdfs

   A core-site.xml kellett átalakítani a következőre:

```
<configuration>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://0.0.0.0:19000</value>
  </property>
  <property>
        <name>hadoop.http.staticuser.user</name>
        <value>hadoop</value>
    </property>
</configuration>
```

20.Core-site.xml

A hdfs-site.xml kellett átalakítani:

```
<configuration>
    <property>
        <name>dfs.replication</name>
        <value>1</value>
    </property>
    <property>
        <name>dfs.namenode.name.dir</name>
        <value>file:///C:/HADOOP/hadoop-3.3.0/data/dfs/namespace_logs</value>
    </property>
    <property>
        <name>dfs.datanode.data.dir</name>
        <value>file:///C:/HADOOP/hadoop-3.3.0/data/dfs/data</value>
    </property>
</configuration>
```

21.hdfs-site.xml

A mapreduce-site.xml átalakítása:

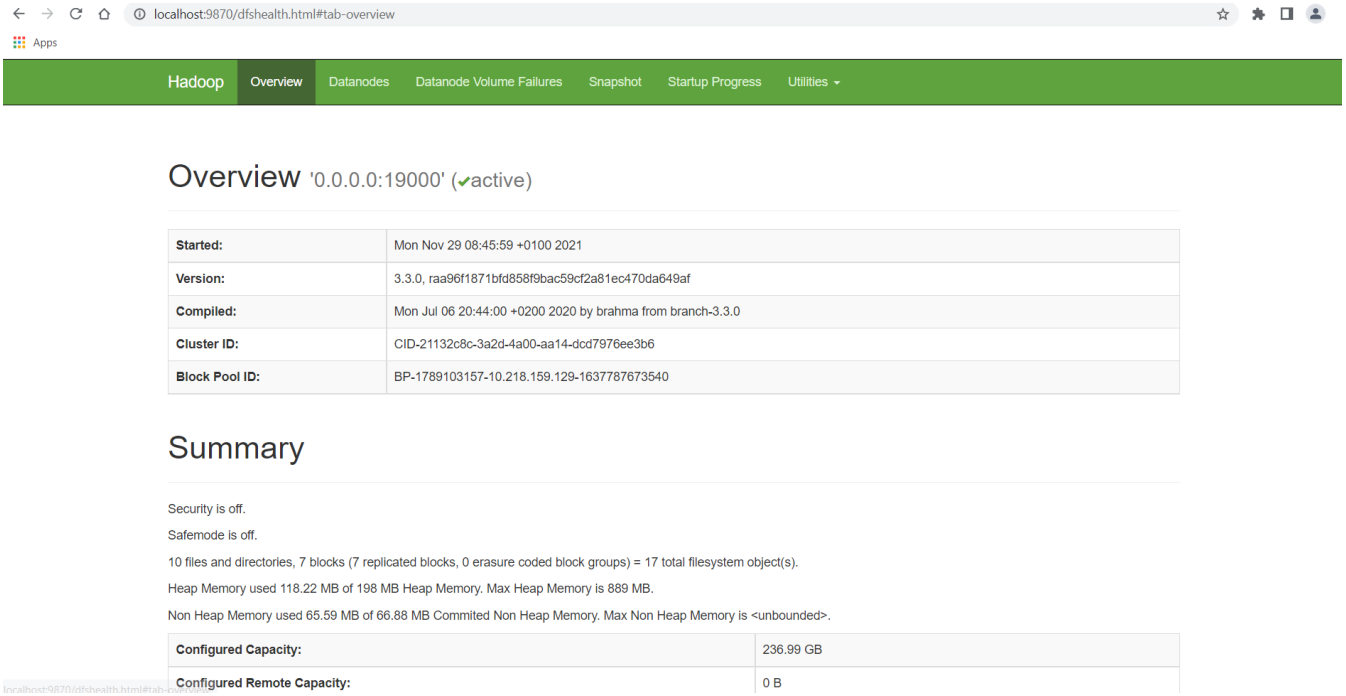
```
<configuration>
    <property>
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
    </property>
    <property>
        <name>mapreduce.application.classpath</name>
        <value>%HADOOP_HOME%/share/hadoop/mapreduce/*,%HADOOP_HOME%/share/hadoop/mapreduce/lib/*,%HADOOP_HOME%/share/hadoop/common/*,%HADOOP_HOME%/share/hadoop/c
    </property>
</configuration>
```

22.mapred-site.xml

A megfelelő beállítások után, Cmd-ből elindítva a szervizeket, el tudtam érni a Hadoop komponenseket. Először a HDFS daemons-t kellet elindítanom a következő parancssal:

```
%HADOOP_HOME%\sbin\start-dfs.cmd
```

Kettő CMD ablak látható, ahol a datanode és a namenode-ról kapunk információt, hogy megfelelően működnek.
A HDFS web portál UI a következő porton érhető el: http://localhost:9870/dfshealth.html#tab-overview.

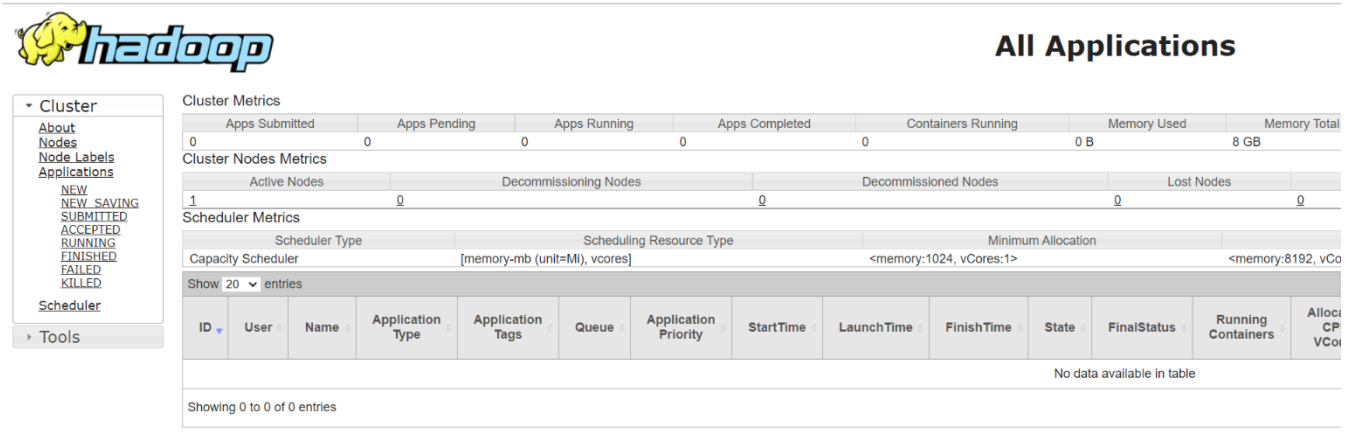


23.Hadoop Web UI

A Yarn Resource Managert adminisztrátorként kellet futtatni cmd-ből a következő paranccsal:

```
%HADOOP_HOME%\sbin\start-yarn.cmd
```

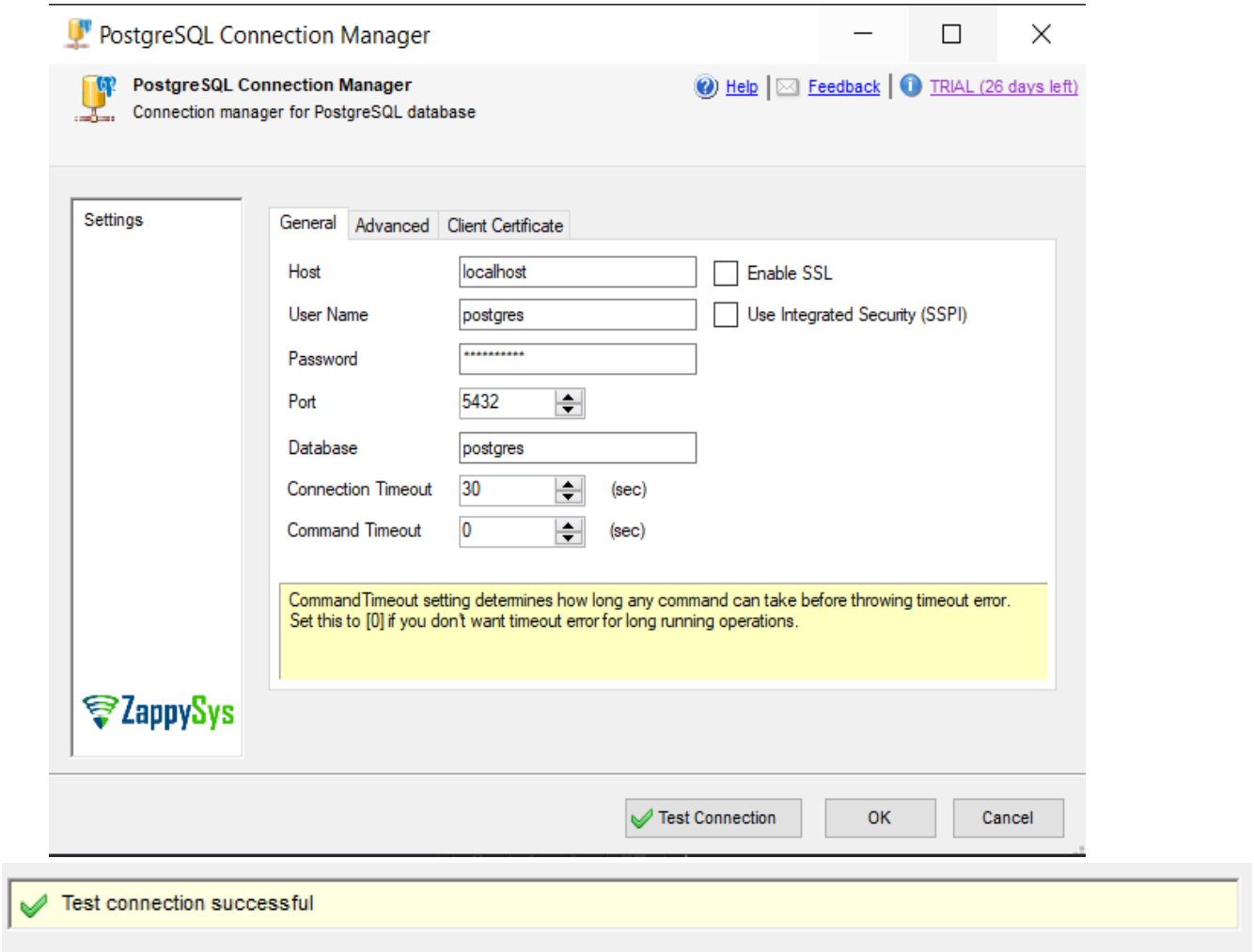
A sikeres indítás után kettő db ablak ugrott fel, az egyik a resource manager node a másik meg resource node.



24.Cluster management

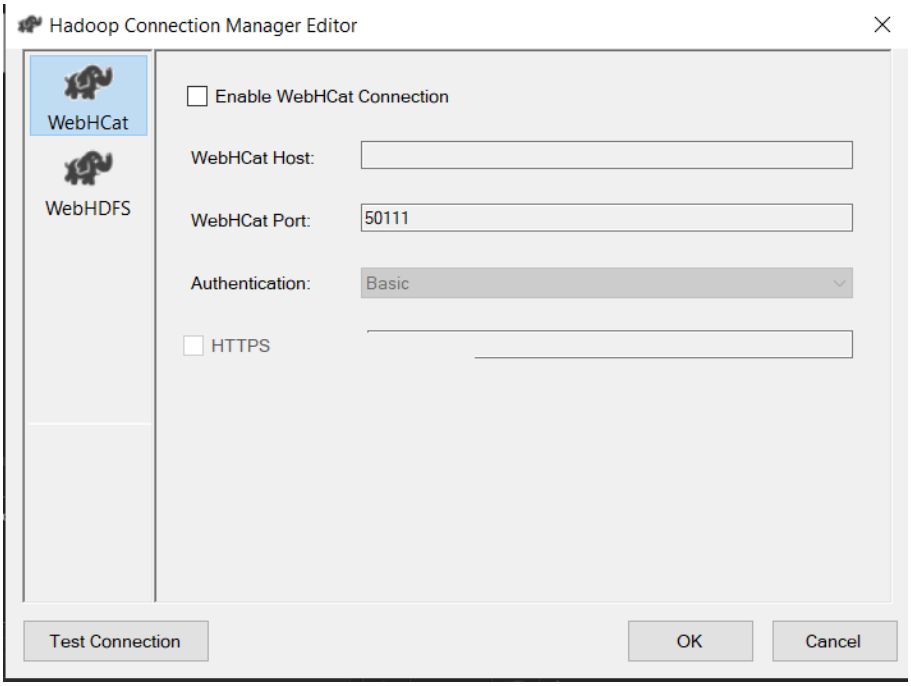
10.5 ETL: SSIS

ETL eszköznek az SSIS választottam, amit Visual Studio 2019 környezetben futtatok.
A cél az, hogy az adatokat a relációs adatbázisból, normalizált formában át töltsük a cél adattárházba.
Data flow task segítségével át tudtam vinni az adatokat relációs adatbázisból HDFS-re.
Ahhoz, hogy létre jöjjön a kapcsolat az adatbázishoz új connection-re volt szükségem.
PostgreSQL connection-t használtam. Megfelelő konfigurálás után a kapcsolat sikeresen létre jött.

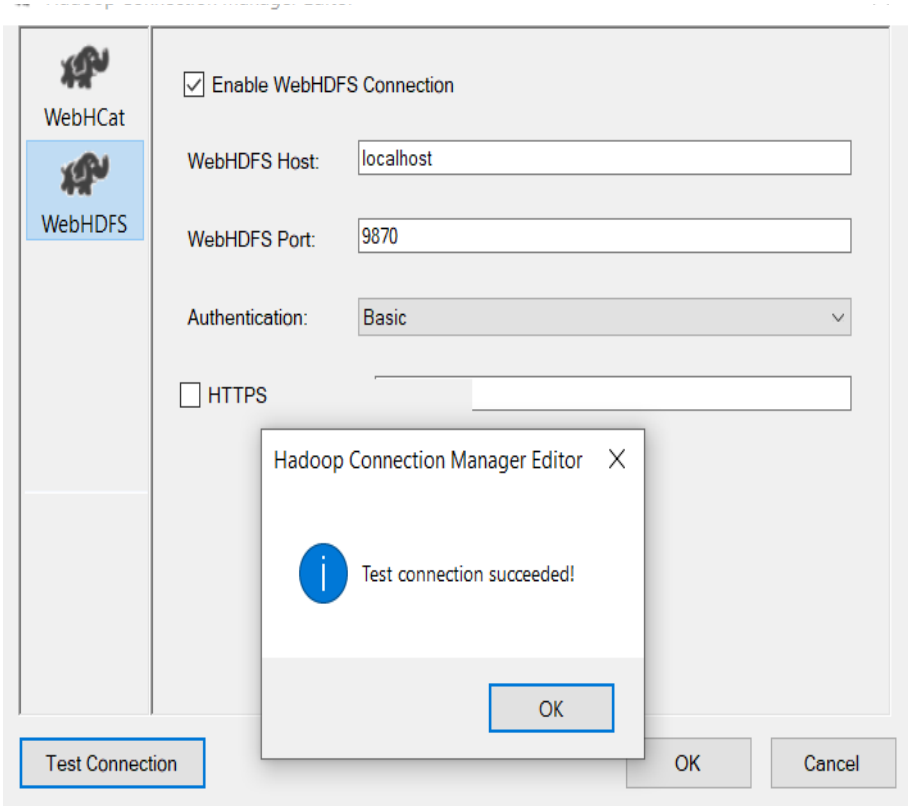


25.PostgreSQL Connection Manager

A Hadoop eléréséhez Hadoop connection kellett létrehoznom.



26.WebHCat connection



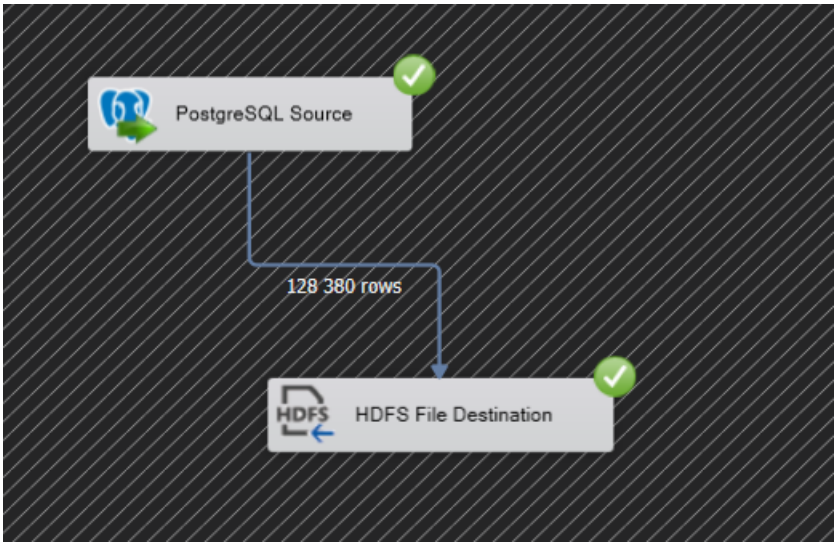
27.WebHDFS connection

A WebHdfs-t a localhost-ra állítottam be. A default 50070-es port nem volt jó, ugyanis nálam a HDFS a 9870-es porton futott. A User_name beállításhoz nem volt elég csak a név, mivel a Hadoopban ahhoz file-kat mozgassunk kellenek jogok. A következő paranccsal tudtam minden jogot adni a felhasználómnak a Hadoop-on belül:
hdfs chmod 777 \tmp.

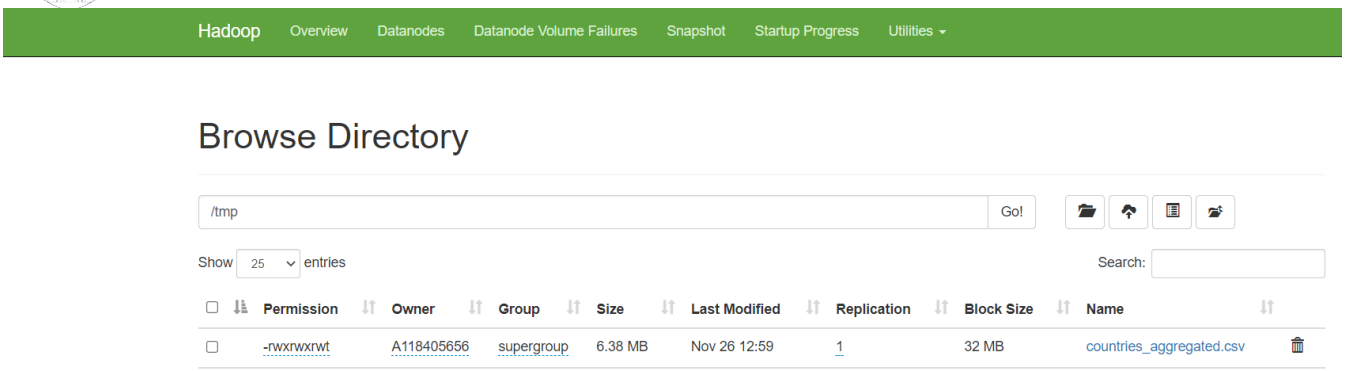
Az első data flow taks a "Get countries_aggregated Data" névre hallgat.
Ennek a tasknak a dolga, hogy az adatokat megfelelő mapping után átvigye hdfs-re.

28.Countries_aggregated Data Flow Task

Az adatok beolvasására PostgreSQL source elem kellett. Első lépésként a létrehozott PostgreSQL connection-t adtam meg. A sikeres kapcsolódás után a táblák láthatóak voltak. Innen a countries_aggregated táblára volt szükség. Az adatokat hdfs-re kellet exportálni, így egy hdfs file destination elemre volt szükségem.
A megfelelő Hadoop connectionnel létre jött a kapcsolat az adatbázis és az adattárház között. A Data Flow taskom lefuttatása után bekerültek az adatok hdfs-re.

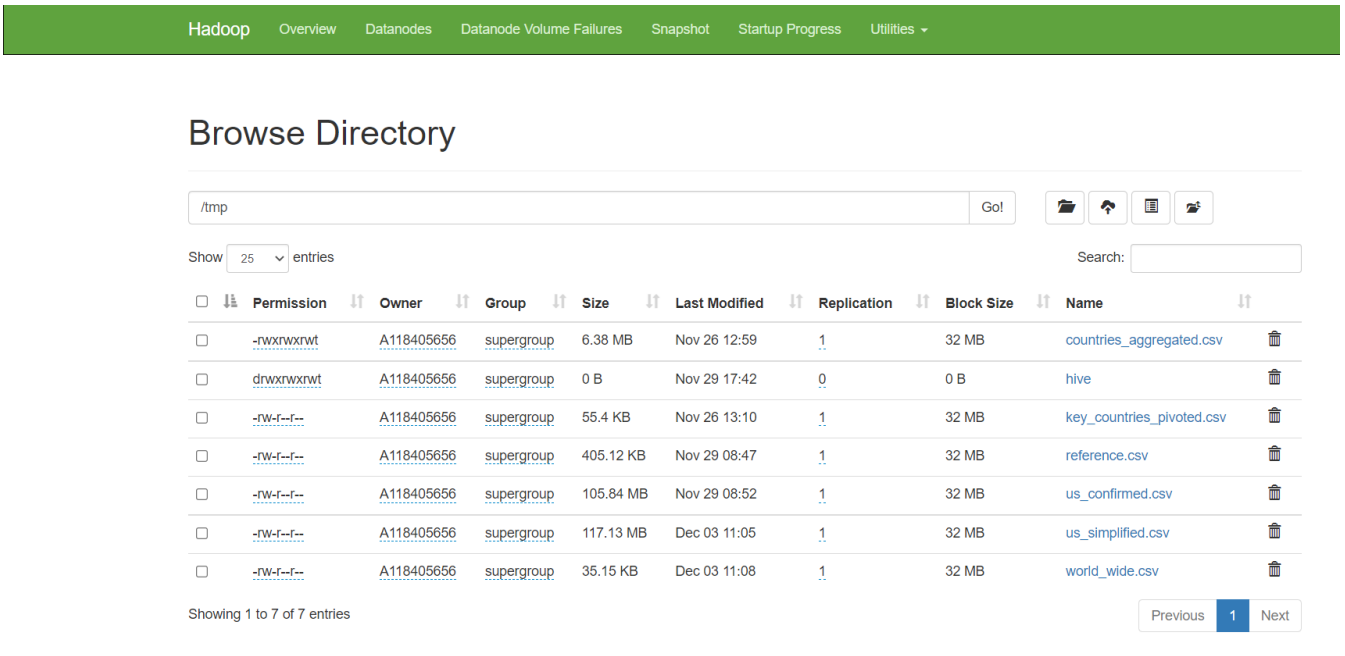


29.Imported rows into HDFS



30.Hadoop browse directory

Az előző lépéseket az összes táblára elvégeztem, és az összes adat bekerült hdfs-re.
A teljes adatállományt a hdfs-en a következő ábra mutatja.



31.Browse Directory with all the files

Az ETL folyamat sikeresen lefutott, minden file megtalálható HDFS-en belül.

10.6: HIVE: telepítés és használata

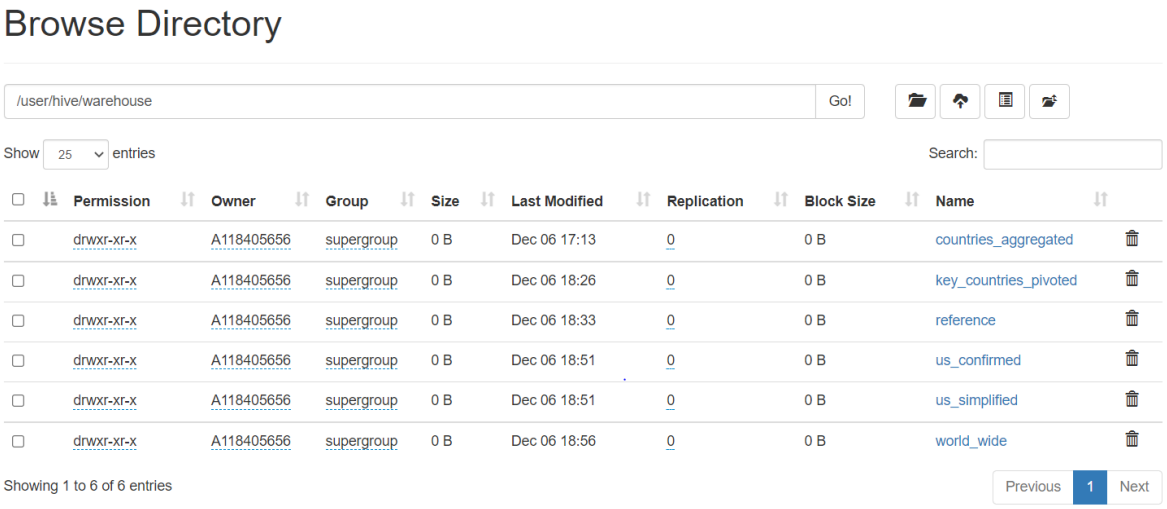
Előfeltételek a Hive futtatásához:

1) **Hadoop telepítése**
   A Hive futtatásához elengedetlen a Hadoop Cluster installálása és futtatása.
2) **Apache Derby**
   Apache Hive-hoz szükséges egy relációs adatbázis telepítése, ahol a metaadatok kerülnek tárolásra, jelen esetben az Apache Derby adatbázis 4-es verziója.
3) **Cygwin**
   Néhány Hive eszköz nem kompatibilis Windows-al (pl:schematool),szükségünk lesz néhány Linux parancsra, hogy futtassuk a Hive-ot. A Cygwin egy olyan eszköz, amivel Windows alatt lehet Linux parancsokat futtatni.

A szervízek indítása:

4) **Hadoop szervizek**
   Apache Hive elindításához, adminisztrátorként megnyitjuk a cmd-t és elindítjuk a megfelelő Hadoop szervizeket.
5) **Derby Network Server**
   A következő paranccsal lehet elindítani localhost-on a Derby szervert:
   C:\hadoop-env\db-derby-10.14.2.0\bin\StartNetworkServer -h 0.0.0.0
6) **Apache Hive indítása**
   Nyissunk terminált adminisztrátorként és lépjünk be a Hive\bin könyvtárba, ahol a következő parancsot futtassuk le: start hive
7) **HiveServer2 indítása**
   Ezzel elindítjuk magát a szervert, ami a következő paranccsal működik:
   hive –service hiveserver2 start

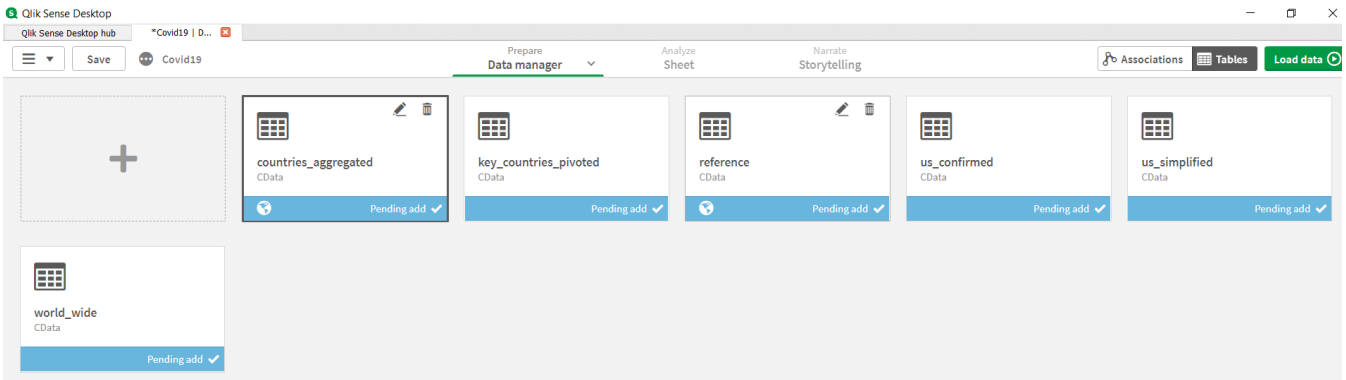
A Hive sikeres indítása után cmd-ből tudunk parancsokat futtatni. Web UI nincs a Hadoop 3.0 verziójához, így mindent terminálból kell futtatni. A megfelelő tábla szerkezetek itt is létre kellett hozni, amit a mellékletekben részleteztem. Az adatokat HDFS-ből importálom a Hive táblákba.
Az alábbi ábrán látható, hogy a táblák sikeresen fel lettek töltve.



32.Hive táblák

10.7: Qlik Sense

Az adatvizualicázióhoz, először létre kellett hoznom Hive ODBC connectiont,hogy a táblák láthatóak legyenek a Qlik Sense számára is. Az adatokat ODBC connection segítségével importáltam be. A kapcsolat létrejötte után a tábláimat az adatokkal együtt sikeresen beimportáltam. A következő ábra mutatja a táblákat a Qlik Sensen belül.

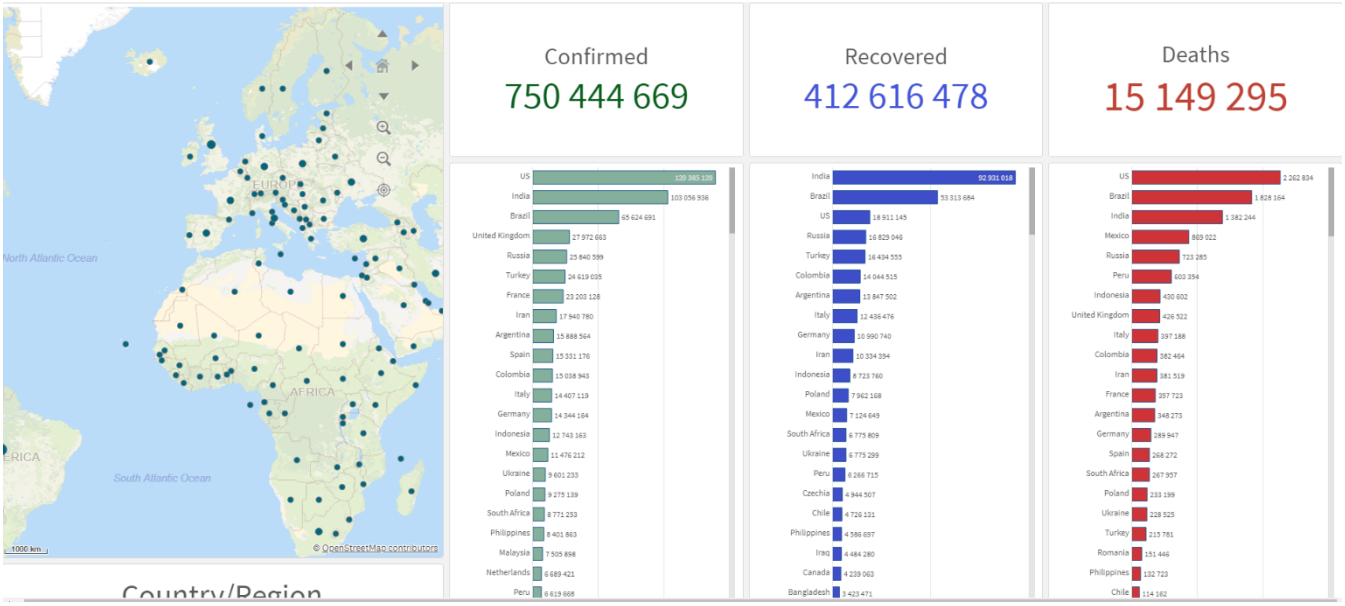


33.Qlik sense Tables

A megfelelő kapcsolatokat létrehoztam és meghatároztam az egyes Use Case-ket a riportokhoz.

10.8: Qlik Sense Reports

Az adatokat át kellett alakítani a megfelelő riportok létrehozásához. Az első átalakítás az volt, hogy felvettem napi adatokat dátumok alapján, ami azt jelenti, hogy kivontam az előző napi adatokból a következő napit. Az USA adatokat tartalmazó táblákat összetudtam konkatenálni, ugyanis számos rekordjuk megegyezett. A kódok, amiken a változtatásokat végeztem a mellékletben találhatóak. Az első riport a fertőzések számát, aktív eseteket, haláleseteket mutatja be részletesen.

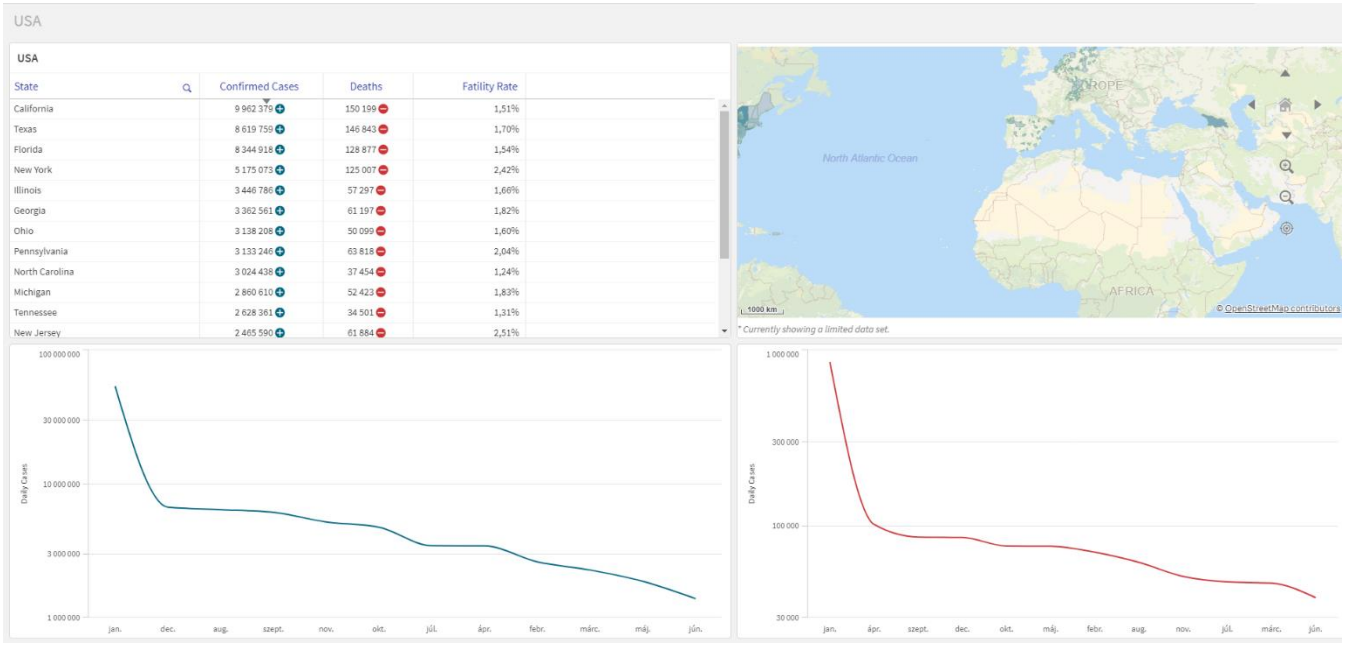


34.Information about infected counties

Az ábrán látható felhasznált eszközök jól mutatják be, hogy alakulnak a fertőzések, aktív esetek, halálozások száma. A bal felső sarokban látható ábra reprezentálja táblázat formában magyarországi eseteket. A táblázaton belül említést teszek a halál, aktív és meggyógyult esetek számáról.
A jobb felső sarokban egy KPI látható, ami a halálozási rátát mutatja meg.
Az alsó diagramok Line Chart-ok amik hónapokra bontva mutatja be a fent említtetett eseteket.

35.Hungary

Az utolsó dashboard az Egyesült Államokat mutatja be részletesen.
A bal felső sarokban KPI objektumot használtam, ahol az összes megbetegedés száma látható az Egyesült Államokon belül. A bal alsó sarokban P&L Pivot objektumot használtam, ahol államokra bontva mutatja a megfertőzések számát. Középen egy térkép segítségével államokra beállítottam, ahol a színek úgy változnak ahogy nőnek a megbetegedések száma. Alatta lévő objektum egy Line Chart ami dátum szerint mutatja a fertőzéseket. A jobb oldali diagram ugyanúgy funkcionál, mint a bal oldali csak a haláleseteket elemzi.



36.Information about USA

11.Továbbfejlesztési lehetőségek

Az informatikai rendszerek nem tökéletesek ebből adódóan, mint minden rendszert tovább lehet fejleszteni. Az elkészült programot is lehet fejleszteni. Az alkalmazást tovább fejlesztési lehetőségei a következők lennének:

- Microsoft Azure környezetbe az alkalmazást fejleszteni.
- Qlik Sense Desktop helyett, felhőbe tárolni az adatokat és a riportokat.
- Az adatmodell újra definiálása egy teljesen új sémába.

12. Összefoglalás

A szakdolgozatom alapjául a 2019-ben kitört korona vírus járvány adta az ötletet.
A járvány a mindennapi életünkben ott van, ezért gyakori téma az emberek között.
A feladatom adattárházba adat betöltés volt és ahhoz riportok, analizálás készítés volt.
Az adatokat egy relációs adatbázisban tároltam, a választás a PostgreSQL-re esett. Könnyen kezelhető és tapasztalatom is volt a rendszerrel. Az adatokat CSV file forrásból importáltam a relációs adatbázisomba. Az adatok exportálására egy ETL (extract,transform,load) eszközt választottam, a választásom a SSIS -re esett mivel iskolai tanulmányaim alatt megismerkedtem vele. Adattárház technológiának Hadoop-ot választottam mivel könnyen skálázható. Az adatok HDFS-re kerültek betöltésre, ami a Hadoop-nak a blokk alapú fájlrendszere. Az adattárházon belül a megfelelő tábla szerkezetek létrehozásáért a Hive felelelt. A táblákba való adatbetöltés HDFS-en tárolt fájlokból történt. Az adatvizualizációs eszköznek a Qlik Sense Desktop verziót használtam a munkálataim során. Az eszközbe való adat betöltéshez ODBC connecitont választottam, ami a kapcsolatért felelt a Hive és a Qlik Sense között.
Az adat betöltés után részletes jelentéseket hoztam létre a korona vírus adatokból, ami egy mélyeb betekintést ad a vírus helyzetéről világszerte.
A felhasznált technológiák során mélyebb betekintést kaptam az adatmodellezésbe, Hadoop mint adattárház és az ETL eszközök világába.
A szakdolgozatom során rengeteg új tapasztalatot szereztem, amit később a munkáim során könnyedén tudok hasznosítani.

13.Summary

My thesis was inspired by the crown virus outbreak in 2019.
The epidemic is in our daily lives, so it is a common topic among people.
My assignment was to load data into a data warehouse and to report and analyse it.
The data was stored in a relational database, PostgreSQL was chosen. It was easy to use and I had experience with the system. I imported the data from a CSV file source into my relational database. To export the data I chose an ETL (extract,transform,load) tool, I chose SSIS because I had been familiar with it during my school studies. I chose Hadoop as the data warehouse technology as it is easily scalable. The data
was loaded on HDFS, which is the block-based file system of Hadoop. Hive was responsible for creating the appropriate table structures within the data warehouse. Data loading into the tables was done from files stored on HDFS. I used the Qlik Sense Desktop version as the data visualization tool during my work. For loading data into the tool, I chose ODBC conneciton, which was responsible for the connection between Hive and Qlik Sense.
Once the data was loaded, I created detailed reports from the crown virus data, which gives a deeper insight into the virus situation worldwide.
The technologies used gave me a deeper insight into data modelling, Hadoop as a data warehouse and the world of ETL tools.
During my thesis I gained a lot of new experience that I can easily use in my future work.

Források, irodalmi hivatkozások

[1] S. J. Alsunaidi *és mtsai.*, „Applications of Big Data Analytics to Control COVID-19 Pandemic”, *Sensors*, köt. 21, sz. 7, Art. sz. 7, jan. 2021, doi: 10.3390/s21072282.
[2] W. G. dos Santos, „Natural history of COVID-19 and current knowledge on treatment therapeutic options”, *Biomed Pharmacother*, köt. 129, o. 110493, szept. 2020, doi: 10.1016/j.biopha.2020.110493.
[3] D. S. W. Ting, L. Carin, V. Dzau, és T. Y. Wong, „Digital technology and COVID-19”, *Nature Medicine*, köt. 26, sz. 4, Art. sz. 4, ápr. 2020, doi: 10.1038/s41591-020-0824-5.
[4] „Real-time Covid 19 Data | Kaggle”. https://www.kaggle.com/gauravduttakiit/covid-19 (elérés máj. 10, 2021).
[5] ZoinerTejada, „Big Data-architektúrák - Azure Architecture Center”. https://docs.microsoft.com/hu-hu/azure/architecture/data-guide/big-data/ (elérés máj. 12, 2021).
[6] Tamás H. és Zoltán P., „Cseh Gábor Kócsó Balázs Váradi Szabolcs”, o. 58.
[7] „Adattárház összefoglaló”. http://scs.web.elte.hu/Work/DW/adattarhazak.htm#11 (elérés máj. 13, 2021).
[8] „Data Visualization: How to Make Sense of Big Data - Talend”, *Talend Real-Time Open Source Data Integration Software*. https://www.talend.com/resources/what-is-data-visualization/ (elérés máj. 13, 2021).
[9] „Running a Data Warehouse on PostgreSQL | Severalnines”. https://severalnines.com/database-blog/running-data-warehouse-postgresql (elérés máj. 14, 2021).
[10] „What is a Data Visualization?” https://www.oracle.com/business-analytics/what-is-data-visualization/ (elérés máj. 13, 2021).
[11] „https://datandroll.hu/2018/05/24/mi-is-az-a-qlik-sense/

Mellékletek

```
create table key_countries_pivoted(date date,
 china int,
 us int,
united_kingdom int,
italy int,
  france int,
   germany int,
 spain int,
  iran int);
create table reference(
        uui int,
        iso2 varchar(10),
        iso3 varchar(10),
        fips int,
        admin2 varchar(100),
        province_state varchar(100),
        country_region varchar(100),
        lat float,
        long_ float,
        combined_key varchar(100),
        population int,
        uuid int,
        code3 int);

create table us_confirmed(
        Country_name varchar(50),
        date date,
        Case int,
        Country/Region varchar(50),
        Province/State varchar(90)
);

create table us_simplified(
        date date,
        admin2 varchar(50),
        Province/State varchar(50),
        confirmed int,
        deaths int,
        Country/Region varchar(50)
```

```
);

Create table world_wide(
        date date,
        confirmed int,
        recovered int,
        deaths int,
        Increase rate int
);
```

```
CREATE TABLE IF NOT EXISTS countries_aggregated(`date` DATE ,country String,
confirmed int,recovered int,deaths int)ROW FORMAT
DELIMITED FIELDS TERMINATED BY ';';

LOAD DATA INPATH '/tmp/countries_aggregated.csv' INTO table countries_aggregated;

select * from countries_aggregated;


CREATE TABLE IF NOT EXISTS key_countries_pivoted(`date` DATE ,china String, us String,
united_kingdom String,italy String,france String,germany String,spain String,iran String)ROW
FORMAT
DELIMITED FIELDS TERMINATED BY ';';

LOAD DATA INPATH '/tmp/key_countries_pivoted.csv' INTO table key_countries_pivoted;

select * from key_countries_pivoted;

CREATE TABLE IF NOT EXISTS reference(uui int ,iso2 String, iso3 String, fips String,admin2
String,province_state String,country_region String,lat double,long_ double,combined_key
String,population int,uid int,code3 int)ROW FORMAT
DELIMITED FIELDS TERMINATED BY ';';

LOAD DATA INPATH '/tmp/reference.csv' INTO table reference;

select * from reference;



CREATE TABLE IF NOT EXISTS us_confirmed(country_name String,`date` DATE ,cases int,
country_region String,province_state String)ROW FORMAT
DELIMITED FIELDS TERMINATED BY ';';

LOAD DATA INPATH '/tmp/us_confirmed.csv' INTO table us_confirmed;

select * from us_confirmed;




CREATE TABLE IF NOT EXISTS us_simplified(`date` DATE ,admin2 String,province_state String,
confirmed int,deaths int , country_region String)ROW FORMAT
DELIMITED FIELDS TERMINATED BY ';';
```

```
LOAD DATA INPATH '/tmp/us_simplified.csv' INTO table us_simplified;


CREATE TABLE IF NOT EXISTS world_wide(`date` DATE ,confirmed int,recovered int,deaths int ,
increase_rate int)ROW FORMAT
DELIMITED FIELDS TERMINATED BY ';';


LOAD DATA INPATH '/tmp/world_wide.csv' INTO table world_wide;
```

```
set vPath = 'lib://QVD';

LIB CONNECT TO 'CData ApacheHive Sys';

//_____________________ Create table Countries_Aggregated
CountriesAggregated:
LOAD `date` as CountriesAggregatedDate,
    country,
    confirmed as CountriesAggregatedConfirmed,
    recovered as CountriesAggregatedRecovered,
    deaths as CountriesAggregatedDeaths;
SQL SELECT `date`,
    country,
    confirmed,
    recovered,
    deaths
FROM CData.`default`.`countries_aggregated`;

//_____________________ Store CSV-s into QVD-s.
store CountriesAggregated into ['$(vPath)/Countries_Aggregated.qvd'](qvd);

//_____________________ Create table KeyCountriesAggregated
KeyCountriesAggregated:
LOAD `date` as KeyCountriesAggregatedDate,
    china,
    us,
    `united_kingdom`,
    italy,
    france,
    germany,
    spain,
    iran;
SQL SELECT `date`,
    china,
    us,
    `united_kingdom`,
    italy,
    france,
    germany,
    spain,
    iran
FROM CData.`default`.`key_countries_pivoted`;
```

```
//____________________ Store CSV-s into QVD-s.
store KeyCountriesAggregated into ['$(vPath)/KeyCountriesAggregated.qvd'](qvd);

LIB CONNECT TO 'CData ApacheHive Sys';

//_______ Create table reference
Reference:
LOAD uui,
    iso2,
    iso3,
    fips,
    admin2,
    `province_state`as ReferenceProvinceState,
    `country_region` as ReferenceCountryRegion,
    lat,
    `long_`,
    `combined_key`,
    population,
    uid,
    code3;
SQL SELECT uui,
    iso2,
    iso3,
    fips,
    admin2,
    `province_state`,
    `country_region`,
    lat,
    `long_`,
    `combined_key`,
    population,
    uid,
    code3
FROM CData.`default`.reference;

//____________________ Store CSV-s into QVD-s.
store Reference into ['$(vPath)/Reference.qvd'](qvd);

LIB CONNECT TO 'CData ApacheHive Sys';

//_________ Create table UsConfirmed
UsConfirmed:
LOAD `country_name`,
```

```
    `date` as UsConfirmedDate,
    cases,
    `country_region` as UsConfirmedCountryRegion,
    `province_state` as UsConfirmedProvinceState;
SQL SELECT `country_name`,
    `date`,
    cases,
    `country_region`,
    `province_state`
FROM CData.`default`.`us_confirmed`;

//____________________ Store CSV-s into QVD-s.
store UsConfirmed into ['$(vPath)/UsConfirmed.qvd'](qvd);

LIB CONNECT TO 'CData ApacheHive Sys';

//__________ Create table UsSimplified
UsSimplified:
LOAD `date`as UsSimplifiedDate,
    admin2,
    `province_state` as UsSimplifiedProvinceState,
    confirmed as UsSimplifiedConfirmed,
    deaths as UsSimplifiedDeaths,
    `country_region` as UsSimplifiedCountryRegion;
SQL SELECT `date`,
    admin2,
    `province_state`,
    confirmed,
    deaths,
    `country_region`
FROM CData.`default`.`us_simplified`;

//____________________ Store CSV-s into QVD-s.
store UsSimplified into ['$(vPath)/UsSimplified.qvd'](qvd);

LIB CONNECT TO 'CData ApacheHive Sys';

//__________ Create table WorldWide
WorldWide:
LOAD `date` as WorldWideDate,
    confirmed as WorldWideConfirmed,
    recovered as WorldWideRecovered,
    deaths as WorldWideDeaths,
```

```
   `increase_rate` as WorldWideIncreaseRate;
SQL SELECT `date`,
   confirmed,
   recovered,
   deaths,
   `increase_rate`
FROM CData.`default`.`world_wide`;

//____________________ Store CSV-s into QVD-s.
store WorldWide into ['$(vPath)/WorldWide.qvd'](qvd);


// ============================================================

// Master Calendar Generation Script

// ============================================================

LET vMinDate = Num(MakeDate(2020,1,1));

LET vMaxDate = Num(MakeDate(2021,12,31));

LET vNumberOfDays = Floor($(vMaxDate)) - Floor($(vMinDate)) + 1;

Calendar:

LOAD Distinct



  Num(Floor(Date)) as DateKey,

  Date(Floor(Date)) as Date,

  Month(Date) as Month,

  Date(MonthStart(Date), 'MMM YYYY') as [Month and Year],

  Day(Date) as Day,

  Year(YearStart(Date)) AS Year,

  'Q'& Ceil(Month(Date)/3) AS Quarter,
```

```
  Week(Date) as Week;

LOAD

  Date(Num($(vMaxDate)) - RecNo() + 1) as Date

AutoGenerate($(vNumberOfDays));

CountriesAggregated:
LOAD
    CountriesAggregatedDate as [CountriesAggregated date],
    CountriesAggregatedDate as DateKey,
    country,
    CountriesAggregatedConfirmed as [CountriesAggregated confirmed],
          CountriesAggregatedConfirmed-Previous(CountriesAggregatedConfirmed) as
[CountriesAggregatedConfirmed daily confirmed],
    CountriesAggregatedRecovered as [CountriesAggregated recovered],
    CountriesAggregatedRecovered-Previous(CountriesAggregatedRecovered) as
[CountriesAggregatedRecovered daily recovered],
    CountriesAggregatedDeaths as [CountriesAggregated deaths],
          CountriesAggregatedDeaths-Previous(CountriesAggregatedDeaths) as
[CountriesAggregatedDeaths daily deaths]
FROM [lib://QVD/Countries_Aggregated.qvd]
(qvd);

store CountriesAggregated into ['$(vPath)/CountriesAggregated.qvd'](qvd);

KeyCountriesAggregated:
LOAD
    KeyCountriesAggregatedDate as [KeyCountriesAggregated date],
    KeyCountriesAggregatedDate as DateKey,
    china,
    china-Previous(china) as [daily china cases],
    us,
    us-Previous(us) as [daily us cases],
    united_kingdom as [united kingdom],
    united_kingdom-Previous(united_kingdom) as [daily united kingdom cases],
    italy,
    italy-Previous(italy) as [daily italy cases],
    france,
    france-Previous(france) as [daily france cases],
    germany,
    germany-Previous(germany) as [daily germany cases],
```

```
    spain,
    spain-Previous(spain) as [daily spain cases],
    iran,
    iran-Previous(iran) as [daily iran cases]
FROM [lib://QVD/KeyCountriesAggregated.qvd]
(qvd);

store KeyCountriesAggregated into ['$(vPath)/KeyCountriesAggregated.qvd'](qvd);

Reference:
LOAD
    uui,
    iso2,
    iso3,
    fips,
    admin2,
    ReferenceProvinceState as [Reference provinceState],
    ReferenceCountryRegion as [Reference countryRegion],
    lat,
    long_,
    combined_key,
    population,
    uid,
    code3
FROM [lib://QVD/Reference.qvd]
(qvd);

store Reference into ['$(vPath)/Reference.qvd'](qvd);


UsConfirmed:
LOAD
    country_name as [Country name],
    UsConfirmedDate as [UsConfirmed date],
    UsConfirmedDate as DateKey,
    cases,
        cases-Previous(cases) as [daily cases],
    UsConfirmedCountryRegion as [UsConfirmed countryRegion],
    UsConfirmedProvinceState as [UsConfirmed provinceState]
FROM [lib://QVD/UsConfirmed.qvd]
(qvd);

store UsConfirmed into ['$(vPath)/UsConfirmed.qvd'](qvd);
```

```
UsSimplified:
LOAD
   UsSimplifiedDate as [UsSimplified date],
   UsSimplifiedDate as DateKey,
   admin2,
   UsSimplifiedProvinceState as [UsSimplified provinceState],
   UsSimplifiedConfirmed as [UsSimplified confirmed],
   UsSimplifiedConfirmed-Previous(UsSimplifiedConfirmed) as [UsSimplifiedConfirmed daily
confirmed],
   UsSimplifiedDeaths as [UsSimplified deaths],
   UsSimplifiedDeaths-Previous(UsSimplifiedDeaths) as [UsSimplifiedDeaths daily deaths],
   UsSimplifiedCountryRegion as [UsSimplified countryRegion]
FROM [lib://QVD/UsSimplified.qvd]
(qvd);

store UsSimplified into ['$(vPath)/UsSimplified.qvd'](qvd);

WorldWide:
LOAD
   WorldWideDate as [WorldWide date],
   WorldWideDate as DateKey,
   WorldWideConfirmed as [WorldWide confirmed],
        WorldWideConfirmed-Previous(WorldWideConfirmed) as [WorldWideConfirmed daily
confirmed],
   WorldWideRecovered as [WorldWide recovered],
   WorldWideRecovered-Previous(WorldWideRecovered) as [WorldWideRecovered daily
recovered],
   WorldWideDeaths as [WorldWide deaths],
   WorldWideDeaths-Previous(WorldWideDeaths) as [WorldWideDeaths daily deaths],
   WorldWideIncreaseRate as [WorldWide increaseRate]
FROM [lib://QVD/WorldWide.qvd]
(qvd);

store WorldWide into ['$(vPath)/WorldWide.qvd'](qvd);

Countries:
LOAD
   "CountriesAggregated date",
   DateKey,
   country,
   "CountriesAggregated confirmed",
   "CountriesAggregatedConfirmed daily confirmed",
```

```
    "CountriesAggregated recovered",
    "CountriesAggregatedRecovered daily recovered",
    "CountriesAggregated deaths",
    "CountriesAggregatedDeaths daily deaths"
FROM [lib://Dashboard qvd/CountriesAggregated.qvd]
(qvd);

KeyCountries:
LOAD
    "KeyCountriesAggregated date",
    DateKey,
    china,
    "daily china cases",
    us,
    "daily us cases",
    "united kingdom",
    "daily united kingdom cases",
    italy,
    "daily italy cases",
    france,
    "daily france cases",
    germany,
    "daily germany cases",
    spain,
    "daily spain cases",
    iran,
    "daily iran cases"
FROM [lib://Dashboard qvd/KeyCountriesAggregated.qvd]
(qvd);


Reference:
LOAD
    uui,
    iso2,
    iso3,
    fips,
    admin2,
    "Reference provinceState",
    "Reference countryRegion",
    lat,
    long_,
    combined_key,
```

```
    population,
    uid,
    code3
FROM [lib://Dashboard qvd/Reference.qvd]
(qvd);

UsConfirmed:
LOAD
    "Country name",
    "UsConfirmed date",
    DateKey,
    cases,
    "daily cases",
    "UsConfirmed countryRegion",
    "UsConfirmed provinceState"
FROM [lib://Dashboard qvd/UsConfirmed.qvd]
(qvd);

UsSimplified:
LOAD
    "UsSimplified date",
    DateKey,
    admin2,
    "UsSimplified provinceState",
    "UsSimplified confirmed",
    "UsSimplifiedConfirmed daily confirmed",
    "UsSimplified deaths",
    "UsSimplifiedDeaths daily deaths",
    "UsSimplified countryRegion"
FROM [lib://Dashboard qvd/UsSimplified.qvd]
(qvd);

WorldWide:
LOAD
    "WorldWide date",
    DateKey,
    "WorldWide confirmed",
    "WorldWideConfirmed daily confirmed",
    "WorldWide recovered",
    "WorldWideRecovered daily recovered",
    "WorldWide deaths",
    "WorldWideDeaths daily deaths",
    "WorldWide increaseRate"
```

```
FROM [lib://Dashboard qvd/WorldWide.qvd]
(qvd);
```