2. feladat

Adatgyűjtés és- előkészítés a lemorzsolódás okainak vizsgálatához

A felmondott előfizetések számának növekedése nyugtalanító helyzetbe hozta a „*Business Talks*" magazin kiadóját. A kialakult helyzet és a kiváltó okok feltárása rávilágított: a lemorzsolódás-vizsgálathoz elengedhetetlen, hogy részletesen elemezzék az értékesítési, illetve előfizetői adatokat. Első lépésben azt kellett tisztázni, hogy milyen adatkörök állnak rendelkezésre a kiadó relációs adatbázisában (amely SQLite technológiát használ) az egyes előfizetőkkel kapcsolatban. Rachel Morgan kérésére az értékesítési csapat összegyűjtötte és rövid leírást adott a rögzített adatok köréről, ezt mutatja be az 1. táblázat.

Tábla	Az egyes mezők (változók) neve	Az egyes mezők (változók) leírása
Subscriptions (Előfizetések)	SubscriptionID [PK]	Előfizetés azonosítója
	STATUS_PORTFOLIO [FK]	Előfizetés állapota a portfóliórendszerben
	STATUS_REASON [FK]	Előfizetés állapotának indoklása a portfóliórendszerben
	TECHNICAL_COMMENCEMENT_DATE	Előfizetés kezdetének technikai dátuma
	VERSION_START_DATE	Előfizetés aktuális verziójának kezdeti dátuma a portfóliórendszerben
	PRODUCT	Termékkód
	PRODUCT_TYPE (Old/New)	Termék típusa
	AGENCY	Előfizetés megkötésében közreműködő ügynökség neve
	REGION	Előfizető lakóhelyének állama
	CITY	Előfizető lakóhelyének városa
	ONLINE_STATUS	Előfizetés értékesítési módja
	WEEKLY_FEE	Előfizetés heti díja dollárban
	ClientID [FK]	Előfizetéshez tartozó ügyfél azonosító kódja
StatusCodes	STATUS_PORTFOLIO [PK]	Előfizetés állapota a portfóliórendszerben (összes lehetséges kód felsorolva)
	Description	Az egyes állapotkódok jelentlésének szöveges leírása
ChurnCodes	STATUS_REASON [PK]	Előfizetés állapotának indoklása a portfóliórendszerben (összes lehetséges kód felsorolva)
	Description	Az egyes indoklás kódok jelentlésének szöveges leírása
Client (Ügyfél)	ClientID [PK]	Ügyfélazonosító kód
	BIRTH_DATE	Ügyfél születésnapja
	Sex	Ügyfél bejegyzett biológiai neme

1. táblázat: Előfizetői adatok köre
Jelmagyarázat: PK = Primary Key; FK = Foreign Key

A második lépésben – a mélységi elemzéseket megelőzően – át kell tekinteni, értelmezni kell a rendelkezésre álló adatokat, hogy megfelelően elő lehessen készíteni a lemorzsolódás-vizsgálathoz. A kiadó 2016 februárjától kezdődően rögzítette előfizetői adatait, melyet a *magazine_subscriptions.db* SQLite adatbázis tartalmaz. Az üzleti oldaltól érkező kérdések megválaszolása előtt tekintsük át a rendelkezésre álló adatokat.

2.1 feladat

a) A *Subscriptions* tábla mezőinek adjuk meg az adattípusát! Az adattípusok közül jelen feladatban négyet különböztessünk meg:
   1. Kategorikus (Nominális)
   2. Dátum
   3. Diszkrét számértékű (a megfigyelések számához képest kis értékkészlet)
   4. Folytonos számértékű (a megfigyelések számához képest nagy értékkészlet)

b) Vizsgáljuk meg a TECHNICAL_COMMENCEMENT_DATE, PRODUCT, WEEKLY_FEE és BIRTH_DATE mezők eloszlását!
   - A megadott mezők közül a kategorikus és dátum típusú mezők esetében egyszerűen, megfelelő aggregálási művelettel vizsgáljuk meg az adatok értékkészletét és azok gyakoriságát, a diszkrét és folytonos számértékű mezők esetén használjuk *pandas* data frame-k hisztogram metódusát!
   - Az esetleges anomáliákra próbáljunk meg lehetséges magyarázatokat adni, és a további feladatok megoldása során vizsgáljuk meg mi lehet az esetleges anomáliák valódi oka!

Morgan több kérdés megválaszolását kérte az elemzői csapattól a lemorzsolódás-vizsgálat előkészítéséhez. *Az elemzések során figyelembe kell venni, hogy az adatok lekérdezésének időpontja 2020.03.01.*

2.2 feladat

Számítsák ki az egyes előfizetők életkorát (egészként) a lekérdezés időpontjában, és az eredményt új mezőként adják a *Subscriptions* táblához! A számítás során vegyék figyelembe, hogy az 1970 előtt született előfizetők születési dátuma és egyéb adatai egy régebbi ügyféladat-kezelő rendszerben kerültek rögzítésre.

Az ügyfél kora a *(lekérdezés dátuma − születési dátum)/365.25* formulából kapott eredmény *egészrész*eként legyen megadva!

2.3 feladat

Az előfizetések értékesítése alapvetően az alábbi három formában történik a kiadónál:

- **Offline**: Teljes mértékben személyes értékesítés, hagyományos szerződésaláírással.
- **Online**: Teljes mértékben online értékesítés a weboldalon keresztül. Értékesítő nem vesz részt a folyamatban.
- **Offline - Digital signature**: Személyes értékesítés, de a szerződéskötés digitális aláírással történik.

Az értékesítési osztály nyilvántartásaiban ezt a három értékesítési formát eltérő kódokon tárolják, és az ezek az eltérő kódok jelennek meg az ONLINE_STATUS mezőben.

Az Önök feladata, hogy egy új mezőben (a *Subscriptions* táblán belül) egységesítsék a kódokat úgy, hogy csak a fenti három esetet különböztesse meg az új kódolás!

2.4 feladat

Az online módon értékesített előfizetések esetében az előfizető városa került rögzítésre, mint földrajzi adat, míg az egyéb előfizetések esetében csak az előfizető államát rögzítette a rendszer. Egy új mezőben egységesen adják meg az előfizető államának kétbetűs rövidítését ANSI szabvány szerint.

A megoldáshoz küldő adatforrásként használhatja a Wikipedia megfelelő szócikkeiben szereplő táblázatokat[1] és az Egyesült Államok kormányzatának megfelelő weboldalát:

- https://en.wikipedia.org/wiki/List_of_United_States_cities_by_population
- https://www.ssa.gov/international/coc-docs/states.html

2.5 feladat

A kiadó egyes ügynökségei gyakran jutalomprogramokat hirdetnek az előfizetők körében, melynek keretében a hűséges ügyfelek tablet vagy TV készülékeket kapnak ajándékba. Az ajándékban részesült ügyfelek listáját az egyes ügynökségek manuálisan rögzítik a *RewardProgram.xlsx* fájlban. A múltban az elemzők tapasztalták már, hogy a manuális rögzítés eredményeképpen bizonyos előfizetések ajándékai többször is szerepeltek a listában.[2]

Azonosítsa az esetleges duplikátumokat az ajándékozásban részesülők listáján, és vizsgálja meg, hogy a duplikált rögzítés köthető-e bizonyos ügynökségekhez!

[1] A megoldás során feltételezhetjük, hogy az előfizető városában a lakosság legalább 100 ezer fő. Egyébként az USA-ban a településnevek kezelésében nagyon fontos lenne az állam mellett a megyék és egyéb közigazgatási egységek használata is. Lehetnek azonos településnevek azonos államban, de eltérő megyében.
[2] A megoldás során feltételezhetjük, hogy egy előfizetés csak egyszer kapott ajándékot a vizsgált időszakban.

A duplikátumoktól megtisztított lista[3] alapján adja meg a *Subscriptions* táblában, egy új mezőben, hogy melyik előfizetés milyen ajándékot kapott! Amennyiben egy előfizetés nem részesült még ajándékban, azt „Nincs” kóddal jelölje!

2.6 feladat

Hasonlítsák össze **a nem online módon értékesített előfizetések esetében** (ahol a földrajzi adatok csak régióval és nem várossal adottak → használható az eredeti REGION mező) az alábbi államokat heti összes díjbevétel és előfizetések száma szerint: Texas, Dél-Karolina, Massachusetts, Utah és Louisiana!

a) Amennyiben az összehasonlítás során anomáliákat tapasztalnak, akkor javítsák azokat egy új mezőben a *Subscriptions* táblában!
b) Az a) pont elvégzése után kategorizálják a heti díjbevétel alapján az előfizetéseket: a díj alapján legalacsonyabb 25%-ba eső előfizetések a „Weak” előfizetések, a díj szerinti középső 50% a „Moderate” címkét kapja, míg a díjbevétel szerinti legerősebb negyedben található előfizetések a „Strong” kategóriába kerüljenek!

2.7 feladat

A *Subscriptions* táblában egy új, „Churn_Status” nevű mezőben adják meg, hogy melyik előfizetések kerültek már felmondásra (1-es kód), és melyek vannak még mindig életben (0-s kód)!
A kiadó azokat az ügyfeleket tekinti lemorzsolódott ügyfélnek, akiknek

- az előfizetése nem élő és
- a törlése nem admin törlés, továbbá
- nem haláleset miatt történt a törlés.

Admin törlés azt jelenti, hogy a törlésre adminisztratív hiba miatt került sor. Ilyen esetekben az ügyfelet rossz paraméterekkel rögzítették a rendszerben, ezért az ügyfél törlésre került, majd a jó paraméterekkel újra felvették a rendszerbe.

2.8 feladat

A 7. feladat eredménye alapján számítsuk ki a *Subscriptions* táblában egy új, „Months_Since_Commencement” nevű mezőbe az előfizetés kezdete óta eltelt *teljes hónapok* számát! Az előfizetés kezdetét minden esetben a TECHNICAL_COMMENCEMENT_DATE mező tartalmazza. Élő előfizetés esetében a hónapok számát a lekérdezés dátumáig, törtölt esetben a törlés dátumáig (VERSION_START_DATE) vegye figyelembe!

[3] A duplikált előfizetések közül az első előfordulást tekintheti helyes adattartalmúnak.

Amennyiben az eredményül kapott érték 1-nél kisebb, akkor azt is vegyük 1-nek. Ezek a megkötés után rögtön visszamondott előfizetéseket jelölik. Elemzési szempontból ezeket úgy kezeljük az előbbi a módosítással, hogy az 1. hónapjuk végére már törlődtek is

A kívánt kimutatások elkészítéséhez, illetve a szükséges adatelőkészítési feladatok elvégzéséhez írjanak egy Python scriptet, amely futtatásával az adatelőkészítési lépések bármikor megismételhetők az adatbázistáblákon. Az adatelőkészítési lépések után nyert, kibővített és javított *Subscriptions* táblát egy új táblaként (az eredetit megőrizve) írja vissza a script az SQLite adatbázisba! Fontos, hogy precízen dolgozzanak, hiszen ezt az adattáblát fogjuk felhasználni a lemorzsolódás-vizsgálathoz!

Megoldás során alkalmazható Python csomagok és függvények

Csomagok

- pandas: https://pandas.pydata.org/
- numpy: https://numpy.org/
- sqlite3: https://docs.python.org/3/library/sqlite3.html
- beautifulsoup4: https://www.crummy.com/software/BeautifulSoup/bs4/doc/
- urllib.request: https://docs.python.org/3/library/urllib.request.html#module-urllib.request

2.0 feladat

- sqlite3.connect: https://docs.python.org/3/library/sqlite3.html
- sqlite3.cursor: https://docs.python.org/3/library/sqlite3.html
- sqlite3.execute: https://docs.python.org/3/library/sqlite3.html
- sqlite3.fetchall: https://docs.python.org/3/library/sqlite3.html
- pandas.read_sql_query: https://pandas.pydata.org/docs/reference/api/pandas.read_sql_query.html
- pandas.DataFrame.info: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.info.html
- sqlite3.close: https://docs.python.org/3/library/sqlite3.html
- pandas.DataFrame.copy: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.copy.html

2.1 feladat

- pandas.DataFrame.groupby: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.groupby.html
- pandas.DataFrame.count: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.count.html
- pandas.DataFrame.plot.bar: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.plot.bar.html

- pandas.to_datetime: https://pandas.pydata.org/docs/reference/api/pandas.to_datetime.html
- pandas.Series.dt.to_period: https://pandas.pydata.org/docs/reference/api/pandas.Series.dt.to_period.html
- pandas.DataFrame.plot: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.plot.html
- pandas.DataFrame.hist: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.hist.html

2.2 feladat

- pandas.Series.str.split: https://pandas.pydata.org/docs/reference/api/pandas.Series.str.split.html
- pandas.DataFrame.info: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.info.html
- pandas.to_numeric: https://pandas.pydata.org/docs/reference/api/pandas.to_numeric.html
- pandas.Series.str.replace: https://pandas.pydata.org/docs/reference/api/pandas.Series.str.replace.html
- numpy.where: https://numpy.org/doc/stable/reference/generated/numpy.where.html
- pandas.Series.str.strip: https://pandas.pydata.org/docs/reference/api/pandas.Series.str.strip.html
- pandas.unique: https://pandas.pydata.org/docs/reference/api/pandas.unique.html
- pandas.DataFrame: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html
- pandas.DataFrame.merge: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.merge.html
- pandas.DataFrame.rename: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.rename.html
- pandas.to_datetime: https://pandas.pydata.org/docs/reference/api/pandas.to_datetime.html
- dict: https://www.w3schools.com/python/ref_func_dict.asp
- pandas.Series.dt.days: https://pandas.pydata.org/docs/reference/api/pandas.Series.dt.days.html
- numpy.floor: https://numpy.org/doc/stable/reference/generated/numpy.floor.html
- pandas.DataFrame.astype: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.astype.html

2.3 feladat

- pandas.DataFrame: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html
- pandas.DataFrame.merge: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.merge.html
- pandas.DataFrame.drop: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.drop.html
- pandas.DataFrame.rename: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.rename.html

2.4 feladat

- urllib.request.Request: https://docs.python.org/3/library/urllib.request.html
- urllib.request.urlopen.read: https://docs.python.org/3/library/urllib.request.html
- bs4.BeautifulSoup: https://www.crummy.com/software/BeautifulSoup/bs4/doc/
- pandas.read_html: https://pandas.pydata.org/docs/reference/api/pandas.read_html.html
- pandas.DataFrame.info: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.info.html
- pandas.DataFrame.rename: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.rename.html
- pandas.Series.str.split: https://pandas.pydata.org/docs/reference/api/pandas.Series.str.split.html
- pandas.DataFrame.loc: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.loc.html
- pandas.Series.str.title: https://pandas.pydata.org/docs/reference/api/pandas.Series.str.title.html
- pandas.DataFrame.merge: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.merge.html
- pandas.DataFrame.isna: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.isna.html
- len: https://www.w3schools.com/python/ref_func_len.asp
- pandas.DataFrame.duplicated: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.duplicated.html
- pandas.DataFrame.drop_duplicates: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.drop_duplicates.html
- numpy.where: https://numpy.org/doc/stable/reference/generated/numpy.where.html
- pandas.DataFrame.drop: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.drop.html

2.5 feladat

- pandas.read_excel: https://pandas.pydata.org/docs/reference/api/pandas.read_excel.html
- pandas.DataFrame.duplicated: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.duplicated.html
- pandas.crosstab: https://pandas.pydata.org/docs/reference/api/pandas.crosstab.html
- pandas.DataFrame.drop_duplicates: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.drop_duplicates.html
- pandas.DataFrame.merge: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.merge.html
- pandas.DataFrame.info: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.info.html
- pandas.DataFrame.loc: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.loc.html

2.6 feladat

- pandas.DataFrame.loc: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.loc.html

- pandas.DataFrame.groupby: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.groupby.html
- pandas.DataFrame.agg: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.agg.html
- pandas.DataFrame.count: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.count.html
- numpy.sum: https://numpy.org/doc/stable/reference/generated/numpy.sum.html
- numpy.where: https://numpy.org/doc/stable/reference/generated/numpy.where.html
- pandas.DataFrame.isin: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.isin.html
- pandas.DataFrame.quantile: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.quantile.html
- pandas.cut: https://pandas.pydata.org/docs/reference/api/pandas.cut.html

2.7 feladat

- numpy.where: https://numpy.org/doc/stable/reference/generated/numpy.where.html
- pandas.DataFrame.isin: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.isin.html
- pandas.DataFrame.loc: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.loc.html

2.8 feladat

- numpy.where: https://numpy.org/doc/stable/reference/generated/numpy.where.html
- pandas.to_datetime: https://pandas.pydata.org/docs/reference/api/pandas.to_datetime.html
- numpy.timedelta64: https://numpy.org/doc/stable/reference/arrays.datetime.html
- pandas.DataFrame.astype: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.astype.html
- numpy.maximum: https://numpy.org/doc/stable/reference/generated/numpy.maximum.html

+1 feladat

- pandas.DataFrame.info: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.info.html
- pandas.DataFrame.drop: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.drop.html
- sqlite3.connect: https://docs.python.org/3/library/sqlite3.html
- pandas.DataFrame.to_sql: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.to_sql.html
- sqlite3.commit: https://docs.python.org/3/library/sqlite3.html
- sqlite3.cursor: https://docs.python.org/3/library/sqlite3.html
- sqlite3.execute: https://docs.python.org/3/library/sqlite3.html
- sqlite3.fetchall: https://docs.python.org/3/library/sqlite3.html
- sqlite3.close: https://docs.python.org/3/library/sqlite3.html