Adatbázis 2 pót pót ZH

A feladatsort figyelmesen olvasd el!

A pót pót ZH feladat leírása mellett található egy **auditalo.exe** állomány, amit első lépésként tölts le (azt töltsd le, ami a pót pót ZH feladat kiírásában található).
Hozz létre egy mappát az alábbi néven: ***név_neptun_kód_adatb2_pot_pot_zh***
Ebbe a létrehozott mappába helyezd el az exe fájlt, majd futtasd (kattints rá duplán).
Ha a Windows blokkolná a futtatást, akkor kattints a felugró ablakban a **További információ** linkre, majd a **Futtatás mindenképpen** opcióra, ezután megnyílik egy command window, ezt ne zárd be, a teljes ZH alatt futnia kell.
Létre fog jönni a mappában egy .git és egy audit mappa, ezeket ne töröld és ne módosíts bele!!!!
A ZH feladatokat úgy készítsd, hogy létrehozol minden feladatnak egy-egy sql kiterjesztésű fájlt (pl.: 1_feladat.sql, 2_feladat.sql) a korábban létrehozott mappában (amiben már fut az exe) és ezeket az sql fájlokat nyitod meg a developer-ben és ott írod a feladatok megoldását és ments folyamatosan.
Ha elkészültél minden feladattal, akkor a korábban létrehozott mappát csomagold zip formátumra.
A tömörített állománynak tartalmaznia kell a .git könyvtárat, az audit mappát és amit az auditáló létrehoz, illetve a megoldásokat tartalmazó sql fájlokat (az exe is maradjon benne).

Ha nem fut a ZH megkezdésétől végig az auditáló, akkor az 0 pontos ZH-t eredményez.
Ha a beadott tömörített állományban nem lesz ott a .git és audit mappa, akkor a ZH 0 pontos.
Ha a .git vagy az audit mappában bármilyen módosítás történik, akkor a ZH 0 pontos.
A ZH során sem egymást, sem semmilyen AI-t nem lehet használni, ezek ZH közbeni vagy utólagosan történő észrevétele esetén a ZH 0 pontos.
Órai anyag és internet használható a feladatok megoldása során, viszont teljes blokkok másolása esetén az adott feladat 0 pontot ér.
A szintaktikailag hibás és nem futtatható kód esetén a feladat 0 pontot ér.
Az invalid kód esetén a feladat szintén 0 pontot ér.

A ZH mellett található egy insert.sql segéd fájl, ha a létrehozott táblák szerkezete (név, oszlopnév, oszloptípus) megegyezik az insert.sql-ben találhatóval, akkor megfuttatás után feltölti adatokkal a táblákat.

1. Hozz létre egy MAGAN_KLINIKA sémát, a jelszó legyen *titkosKlinika123* és a sémát használd a fejlesztés további részében.

2. Az adatbázisban orvosokat, pácienseket és a köztük létrejött vizsgálatokat kell tárolni.
Az orvosról kötelezően tárolunk azonosítót, a nevét, a pecsétszámát (egyedi azonosító, szöveges), a szakterületét (pl. Belgyógyász, Sebész) és a vizsgálati alapdíját (szám).
A páciensről kötelezően tároljuk az azonosítót, a nevét, a TAJ számát (egyedi, szöveges), születési dátumát és a lakcímét.
A vizsgálat kapcsolótáblában tároljuk a kapcsolat azonosítóját, az orvos azonosítót, a páciens azonosítót, a vizsgálat dátumát, a diagnózist (szöveges) és a fizetett összeget (ami eltérhet az alapdíjtól).
A táblák azonosítója egyértelműen azonosítson minden rekordot. Ahol az egyik tábla adatait használjuk fel egy másikban, ott össze kell kapcsolni a két táblát. A vizsgálat táblában az orvos, a páciens és a dátum hármasának egyedinek kell lennie (ugyanaz az orvos ugyanazt a beteget

ugyanabban az időpontban nem vizsgálhatja kétszer).
A TAJ szám hossza csak 9 karakter lehet.

3. A táblák azonosítója automatikusan kerüljön kitöltésre, ne kelljen kézzel megadni beszúráskor. Ezen felül minden táblában legyen három audit mező, amelyek automatikusan töltődnek:

- LETREHOZVA: a rekord keletkezésének pontos ideje.
- MODOSITVA: a rekord keletkezésének vagy utolsó módosításának pontos ideje.
- MODOSITO_FELHASZNALO: az adatbázis műveletet végző operációs rendszer szintű felhasználó neve. Az orvos tábla azonosítója 100-tól, a páciensé 1500-től, a vizsgálaté 1-től induljon.

4. Készíts egy nézetet, hogy a recepción dolgozók láthassák a vizsgálatokat, de a TAJ szám adatok védve legyenek, illetve a konkrét árak helyett csak árkategóriákat lássanak.
A nézet tartalmazza a páciens nevét, az orvos nevét és a vizsgálat dátumát, TAJ számot, és az ár kategóriát.
A páciens TAJ számának csak az utolsó 3 karakterét jelenítsd meg, a többi karakter helyett csillagok szerepeljenek (pl.: ******789).
Hozz létre egy oszlopot az alábbi logika szerint:

- Ha a fizetett összeg nincs kitöltve vagy 0, akkor: 'Térítésmentes'.
- Ha az összeg 1 és 50.000 Ft közé esik, akkor: 'Normál díjas'.
- Ha az összeg 50.000 Ft feletti, akkor: 'Kiemelt díjas'.

5. A rendszerben kritikus fontosságú a vizsgálatok módosításainak követése. Hozz létre egy VIZSGALAT_AR_HIST táblát (vizsgálat azonosító, régi ár, új ár, módosítás dátuma). Készíts egy triggert, ami a vizsgálat tábla adatainak módosításakor automatikusan elmenti a rekord azonosítóját, a fizetett összeg régi és új értékét, valamint a módosítás idejét ebbe a táblába.
Csak akkor mentsen, ha volt változás az árban!!!!

6. Készíts egy tárolt eljárást, amely egy adott orvos vizsgálati alapdíját módosítja. A folyamat legyen tranzakcióbiztos és szimuláljon egy hosszú feldolgozási folyamatot:

- Először zárolja az adott orvos rekordját, hogy más ne módosíthassa.
- Várjon 5 másodpercet (altassa a folyamatot).
- Csak ezután hajtsa végre a módosítást. Ha a megadott orvos azonosítóval nem található rekord, a program dobjon egy doctor_not_found_exc kivételt.

7. Készíts egy tárolt eljárást, amely egy dinamikusan összeállított SQL utasítást futtat le. Az eljárás bemeneti paraméterei legyenek egy dátum és egy összeghatár. Az eljárásnak legyen egy kimeneti paramétere is, amelyben visszaadja azon vizsgálatok darabszámát, amelyek a megadott dátum *után* történtek és a fizetett összeg *magasabb* volt, mint a megadott határ.

8. Készíts egy típust és hozzá egy listát. Az mezői legyenek: orvos neve, szakterület, összes bevétel (az orvoshoz tartozó vizsgálatok fizetett összegeinek összesítése).

9. Készíts egy függvényt, amely visszaadja az előző feladatban létrehozott listát, de az alábbi módon összeállítva:

- Külön gyűjtsd ki egy listába a 'Belgyógyász' szakterületű orvosok adatait.
- Külön gyűjtsd ki egy másik listába a 'Sebész' szakterületű orvosok adatait.
- A függvény a két lista összefűzött unióját adja vissza eredményként (tehát egyetlen közös listát, ami tartalmazza mindkét listát).

10. Hozz létre egy tárolt eljárást, ami naplózza a rendszer pillanatnyi állapotát (pl. hány orvos és páciens van jelenleg) egy RENDSZER_LOG táblába (azonosító, dátum, információ). Ezt követően készíts egy Job-ot, amely ezt a tárolt eljárást hívja meg.
Egy külön utasítással manuálisan indítsd el ezt a jobot tesztelési céllal.