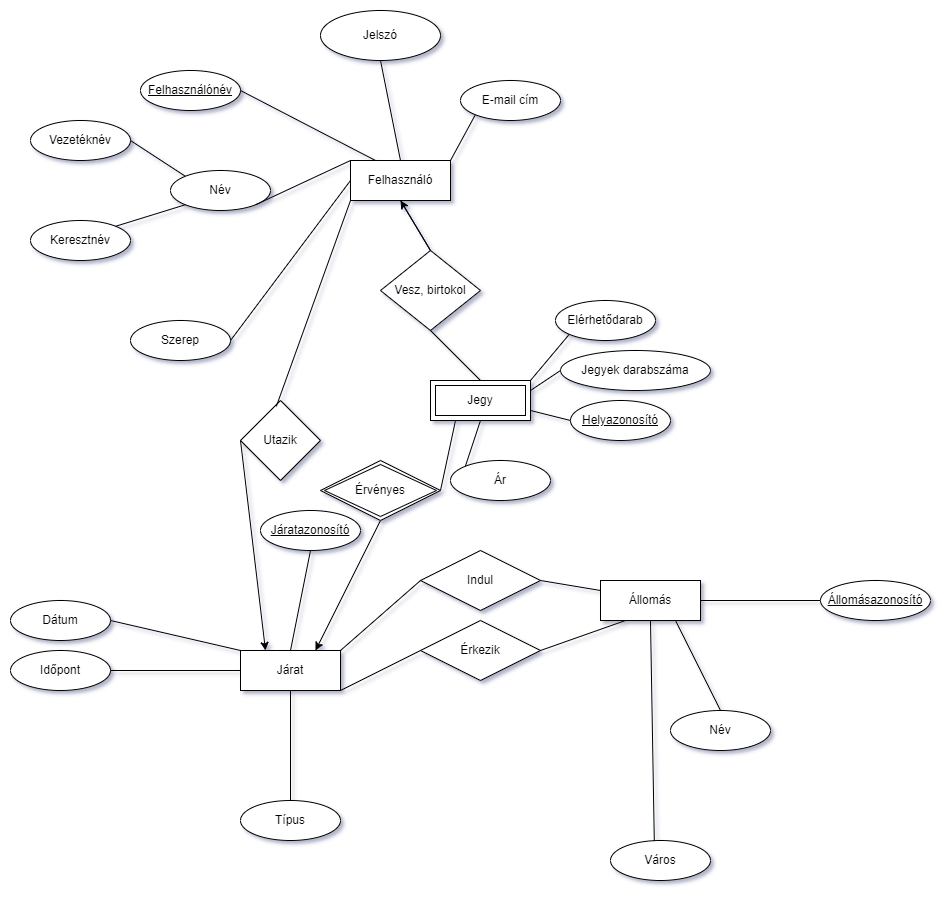
**Adatbázisok projektmunka dokumentáció - Fábián Bernát**

**Busz, vasút, repülő helyfoglalás**

**Adatbázisok kötelező feladat**

Valósítsuk meg különböző típusú közlekedési eszközökre a jegyek adminisztrációját és vásárlását. A felhasználó (vásárló) a regisztráció előtt vendégként csak a járatokat látja, regisztráció után vásárolni tud jegyeket (többet is egyszerre akár). Lehessen szűkíteni úgy, hogy megadunk két állomást, és a közte elérhető járatokat listázza csak, illetve lehessen szűrni csak repülőre, buszra vagy vonatra.

**Egyed-kapcsolat modell**

****

**Felhasználó:**

**Felhasználói szerepek:**

**Vendég:** A vendég felhasználók csak a megtekinthetik az állomásokat és a járatokat, nem vásárolhatnak jegyeket. A vendégnek nincsenek attribútumai, hiszen nem tárolunk el róla semmilyen információt.

**Regisztrált felhasználó:** A regisztrált felhasználók járatokat tekinthetnek meg, jegyeket vásárolhatnak, és különböző szűrési lehetőségekkel használhatják az alkalmazást. A felhasználónévnek és az emailnek is egyedinek kell lennie, ezt regisztációkor ellenőrizzük. A felhasználót a felhasználóneve alapján azonosítjuk. A felhasználónév rövidebb, mint az e-mail cím, ezért érdemes azt használni kulcsnak. A többi adat (jelszó, vezetéknév, keresztnév, szerep, timestamp, is\_logged\_in) megegyezhet több felhasználónál is. A név egy összetett attribútum, ezt 1NF-re hozáskor vezeték- és keresztnévre bontjuk.

**Adminisztátor:** több funkcióhoz fér hozzá, mint egy sima felhasználó.

**Állomás:** Az állomást egyértelműen meghatározza az állomás neve és a város neve, amelyben található. A kulcsa viszont egy egyedi állomásazonosító lesz az egyszerűség kedvéért, ami egy szám.

**Jegy:** A jegy gyenge entitás, a járatazonosító, amely külső kulcs és a helyazonosító együttesen egyértelműen be tudnak azonosítani egy jegyet. Mivel a helyazonosító csak egy egyszerű szám, a járatazonosító is szükséges az aattribútumok beadonosításához.

Feltételezzük, hogy 1 felhasználó 1 időpontban tetszőleges számú jegyet tud venni, azonban 1 járathoz 1 helyre csak 1 jegyet vehet.

**Járat:**  Az időpont és a dátum között az a különbség, hogy míg a dátum csak az év-hónap-nap adatokat tárolja (YYYY-MM-DD formátumban), addig az időpont a óra-perc-másodperc adatokat járol (HH:MM:SS formátumban).

**Felhasználójegyek:** A felhasználó jegyeit nem kell külön entitásként, külön táblában kezelni, mert ez csak több entitás azonosítóinak az egy helyen tárolása lesz.

**Kapcsolatok:**

1 jegy csak 1 felhasználóhoz tartozhat, de 1 felhasználónak több jegye is lehet (1:N kapcsolat)

1 jegy csak 1 járathoz tartozhat. 1 járathoz több jegy és felhasználó tartozhat (1:N kapcsolat)

1 állomáshoz több járat is tartozhat és 1 járathoz több állomás is tartozhat (N:M kapcsolat)

**Relációs adatbázisséma**

Felhasználó (felhasználónév, email, jelszó, vezetéknév, keresztnév, szerep, utolsó belépés időpontja, be van-e jelentkezve)

Járat (járatazonosító, típus, indulóállomás, célállomás, dátum, időpont)

Állomás (állomásazonosító, név, város)

Jegy (*járatazonosító*, helyazonosító, ár, elérhetődarab, jegyek darabszáma, tulajdonos)

Jegyvásárlás(id, járatazonosító, darabszám)

**Normalizálás**

**Az 1NF teljesül.**

**Egy *R(A,F)* relációséma 1. normálformában van, ha az *A* attribútum-halmaz minden eleme atomi, vagyis nem többértékű és nem összetett attribútum.**

* **Az összetett attribútumokat a részattribútumaikkal helyettesítjük.**
* **A többértékű attribútumokat célszerűen új relációsémába vesszük fel, amelyhez külső kulcsként hozzávesszük az őt tartalmazó relációséma kulcsát.**

**Indoklás:**

Minden olyan információt, amire önálló egységként lehet tekinteni, azt szétbontottam.

A felhasználó név összetett attribútumot vezeték- és keresztnévre bontással helyettesítettem. A

dátumot és időpontot is külön kezeljük az adatbázisban.

Többértékű attribútum nem volt a relációsémák között.

**A 2NF teljesül.**

Egy relációséma 2NF-ben van, ha olyan attribútum, amely nem szerepel a relációséma valamely kulcsában, teljesen függ bármely kulcstól.

A sémát felbontjuk Heath tétele szerint, a normálformát sértő függőség mentén.

Ha valamely *K* kulcsra *L ⊂ K* és *L→B* (itt *B* legyen az összes *L*-től függő attribútum halmaza), akkor a sémát felbontjuk az *L→B* függőség szerint. Legyen *C = A – (L U B)*, ekkor az *R(A)* sémát az *R1 (C U L)* és *R2 (L U B)* sémákkal helyettesítjük.

**Indoklás:**

Minden másodlagos attribútum teljesen függ bármely kulcstól, azaz minden attribútum, amely nem szerepel a relációséma egyik kulcsában sem, az teljesen függ bármely kulcstól. Ez még a jegy attribútumaira is igaz, mivel a helyazonosító csak egy egyszerű szám, a járatazonosító is szükséges az aattribútumok beadonosításához.

**A 3NF teljesül.**

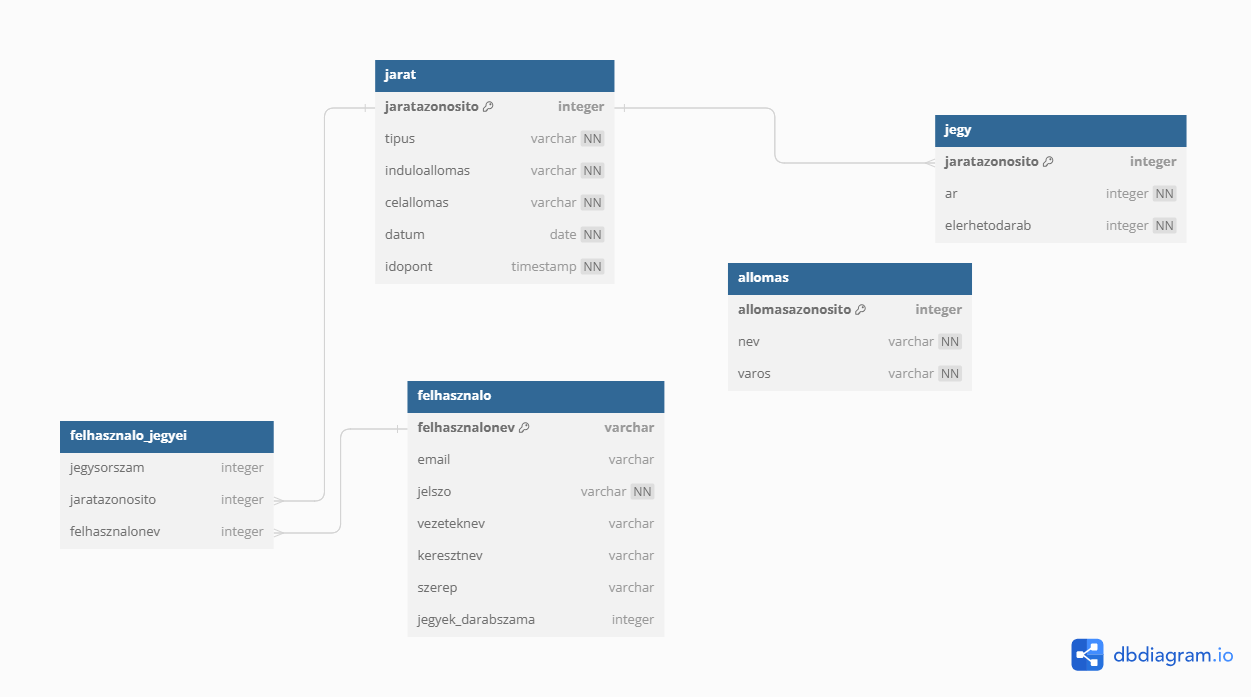
**Ha másodlagos attribútumok egy *B* halmazára és valamely *K* kulcsra *K→Y→B* tranzitív függés fennáll, akkor a sémát felbontjuk Heath tétele szerint az *Y→B* függés mentén.**

***B* legyen az összes *Y*-tól függő attribútum halmaza. Legyen *C = A – (Y U B)*, ekkor az *R(A)* sémát az *R1 (C U Y)* és *R2 (Y U B)* sémákkal helyettesítjük. Heath tétele alapján a felbontás hűséges.**

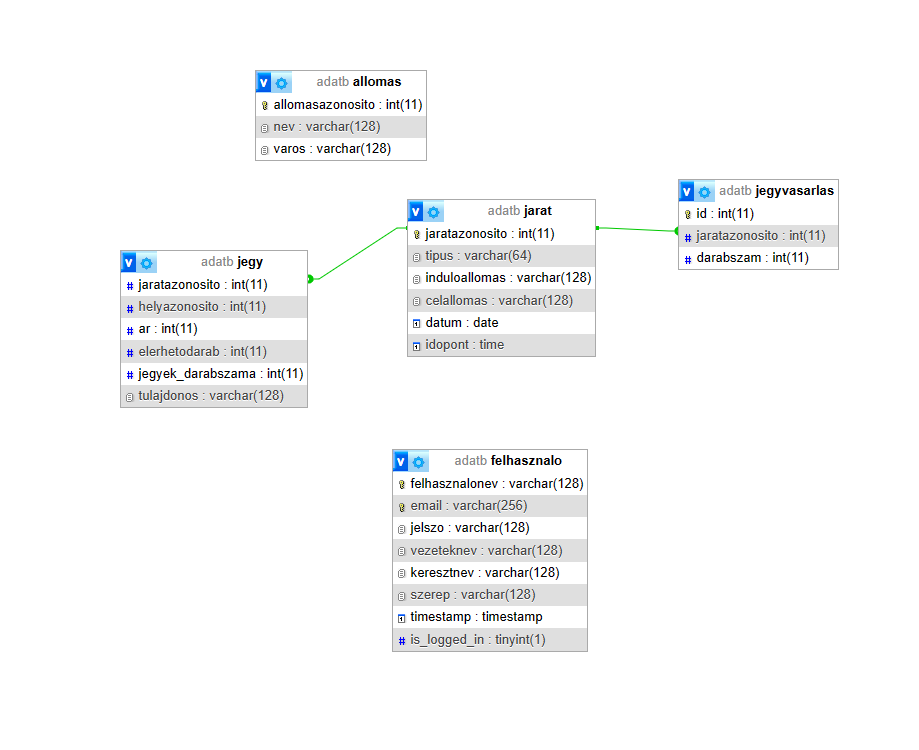
**Indoklás:** Az időpont és a dátum szétbontásával az időpontból sem határozható meg a dátum és fordítva sem.

**Táblatervek**

[**https://dbdiagram.io/d/Adatb\_helyfoglalas\_tablaterv-654e2fdc7d8bbd6465ee8f80**](https://dbdiagram.io/d/Adatb_helyfoglalas_tablaterv-654e2fdc7d8bbd6465ee8f80)

****

**Végleges:**

****

**Összetett lekérdezések**

Az adatbázis létrehozó SQL kódot a C:\XAMPP\htdocs\Adatbazis\_projektmunka elérési úton lévő fájlba mentettem. Ez létrehozza az adatbázist localhoston.

A lekérdezések a C:\XAMPP\htdocs\Adatbazis\_projektmunka\ elérési úton találhatóak meg (a gépemen).

3 példa:

**C:\XAMPP\htdocs\Adatbazis\_projektmunka\includes\list\_tickets\_by\_station\_with\_info.inc.php**

**$sql =** SELECT

j.jaratazonosito,

j.helyazonosito,

j.ar,

j.elerhetodarab,

j.jegyek\_darabszama,

COUNT(\*) AS ticket\_count,

st.nev,

st.varos

FROM jegy j

INNER JOIN jarat jr ON j.jaratazonosito = jr.jaratazonosito

INNER JOIN allomas st ON jr.induloallomas = st.allomasazonosito

GROUP BY j.jaratazonosito, j.helyazonosito”;

**C:\XAMPP\htdocs\Adatbazis\_projektmunka\includes\list\_popular\_routes.inc.php**

**$sql = "SELECT jarat.jaratazonosito, COUNT(\*) AS eladott\_jegyek**

**FROM jegy**

**JOIN jarat ON jegy.jaratazonosito = jarat.jaratazonosito**

**GROUP BY jarat.jaratazonosito**

**ORDER BY eladott\_jegyek DESC**

**";**

**C:\XAMPP\htdocs\Adatbazis\_projektmunka\includes\list\_tickets\_by\_vehicle\_type.inc.php**

**SELECT tipus, COUNT(\*) AS ticket\_count FROM jarat GROUP BY tipus**

**Megvalósítás, funkciók**

Az adatbázissal támogatott alkalmazás a XAMPP szabad és nyílt forrású platformfüggetlen webszerver-szoftvercsomag segítségével lett megvalósítva. A kódokat PHPStorm fejlesztői környezetben, .php, .sql és .html kiterjesztésű kódfájlokban írtam meg, Google Chrome és Microsoft Edge böngészőkben teszteltem . A használt webszerver az Apache, 80-as és 443-as porton. Az elkészített adatbázist a MariaDB (MySQL) , 3306-os porton, <http://localhost/phpmyadmin/> linken értem el.

* Az E-K diagramot a <https://app.diagrams.net/> webalkalmazással készítettem el.
* A táblaterveket a [dbdiagram.io - Database Relationship Diagrams Design Tool](https://dbdiagram.io/home) webalkalmazással készítettem el.
* A fájlokat Githubra is elmentettem, a Repo linkje: <https://github.com/Fabbernat/Adatbazis_projektmunka>

Minta felhasználónév: d

jelszó: d