

TUDOMÁNYOS DIÁKKÖRI DOLGOZAT

**RELÁCIÓS ADATBÁZISOK NORMALIZÁlásÁHOZ HASZNÁLT ALGORITMUSOK AZ OKTATÁSBAN**

|  |  |  |
| --- | --- | --- |
|  | **Szerző(k):** | **Kiss Gergely** |
|  |  | Számítástechnika MSc. szak, I. évf. |

|  |  |  |
| --- | --- | --- |
|  | **Konzulens(ek):** | **dr. Első Konzulens** |
|  |  | egyetemi docens |

**SapientiA Erdélyi magyar tudományEgyetem**

**RELÁCIÓS ADATBÁZISOK NORMALIZÁlásÁHOZ HASZNÁLT ALGORITMUSOK AZ OKTATÁSBAN**

**RELATIONAL DATABASE NORMALIZATION ALGORITHMS IN EDUCATION**

Kiss Gergely

Konzulens(ek):

Dr. Első Konzulens

Kézirat lezárva: 2017 április 12.

**Abstract**

Ide jön a ½-1 oldalas angol nyelvű kivonat.

**Kivonat**

Ide jön a ½-1 oldalas magyar nyelvű kivonat.

Tartalomjegyzék

[1 Bevezető 1](#_Toc99575406)

[2 Irodalom áttekintés 1](#_Toc99575407)

[3 Célkitűzések 1](#_Toc99575408)

[4 Elméleti megalapozás 1](#_Toc99575409)

[4.1 A relációs adatmodell alapvető fogalmai 1](#_Toc99575410)

[4.2 Normálformák 3](#_Toc99575411)

[4.2.1 1NF 3](#_Toc99575412)

[4.2.2 2NF 4](#_Toc99575413)

[4.2.3 3NF 4](#_Toc99575414)

[4.2.4 BCNF 4](#_Toc99575415)

[4.3 Normalizációs algoritmusok 4](#_Toc99575416)

[4.3.1 Szintézis 4](#_Toc99575417)

[4.3.1.1 A szintézis algoritmusa 5](#_Toc99575418)

[4.3.2 Dekompozíció 7](#_Toc99575419)

[4.3.2.1 A dekompozíció algoritmusa 7](#_Toc99575420)

[4.4 Szoftvermodellezési szempontok 9](#_Toc99575421)

[5 Gyakorlati megvalósítás 9](#_Toc99575422)

[6 Eredmények 9](#_Toc99575423)

[7 Tárgyalás 9](#_Toc99575424)

[8 Összefoglalás 9](#_Toc99575425)

[9 Irodalomjegyzék 10](#_Toc99575426)

Ábrajegyzék

# Bevezető

A relációs adatbázisok alapjául szolgáló relációs adatmodell már az 1970-es években sok tudományos munka tárgyát képezte, és az 1990-es évektől kezdve kezdték alkalmazni kommerciális környezetben is (Mogin & Luković, dátum nélk.). Napjainkban a legkülönfélébb vállalkozások is nagy arányban használnak relációs adatbázist valamilyen formában, ezt több felmérés (ScaleGrid.io, 2019) (Ramel, 2015) és jelentés (Loukides, 2022) is bizonyítja. Egy nagy előnye a relációs adatbázisoknak, hogy a relációs adatmodell szilárd matematikai alapokra épül. Ezek az alapok többek között a halmazelméletet és a matematikai relációkat foglalják magukban, ami engedélyezi a relációs algebra és kalkulus műveleteinek a használatát (Mogin & Luković, dátum nélk.).

Annak érdekében, hogy egy relációs adatbázison el tudjunk végezni bizonyos relációs műveleteket, és pontos eredményekhez jussunk, elengedhetetlen az a feltétel, hogy az adatainkat veszteségmentesen egyesíthető relációkba szervezzük. Veszteséges egyesítésnél adatok tűnhetnek el, vagy megjelenhetnek további téves adatok. Ezen probléma kiküszöböléséhez adatbázis normalizálást kell végrehajtanunk, amivel a relációs sémákat átszervezzük olyan formába, amely veszteségmentes egyesítést eredményez. Ezen normalizálási műveletek sikeres elvégzéséhez a dolgozat két algoritmust is bemutat.

Jelenleg az Újvidéki Egyetem munkatársa vagyok, a Műszaki Tudományok Karán végzek tanársegédi feladatokat. Ezen a karon több szakirányon is folyik relációs adatbázistervezéssel foglalkozó tárgy. A tárgy neve Adatbázisok 2 (Műszaki Tudományok Kara, Újvidék, 2021), melynek keretén belül előadjuk az említett normalizálási algoritmusokat is. Az algoritmusokat feladatok kíséretében dolgozzuk fel kézileg és ugyanígy papíron történik a hallgatók vizsgáztatása is. Egy feladatlap elkészítése, megoldáskulcs ellenőrzése, majd a későbbi hallgatók által kitöltött feladatlapok átnézése potenciálisan sok időt felemészt, valamint nagy felelőséggel is jár. A dekompozíció normalizálási algoritmusa választási lehetőség elé állítja az alanyt, vagyis különböző útvonalakon juthat el a hallgató a helyes eredményig. Ez az interaktív mozzanat tovább bonyolítja a megoldott feladatok kiértékelését – adott esetben a részeredmények helyességének a megállapítását.

Annak érdekében, hogy enyhítsünk a tanársegédekre helyezedő nehézségeken, egy szoftvert fejlesztenénk ki, mely képes pontosan elvégezni a relációs adatbázis normalizálási feladatait, lehetővé téve az interaktív lépések végrehajtását. A szoftver emellett még megkönnyítené a feladatsorok megadási módját is, ezzel is felgyorsíthatjuk a feladatok kidolgozásának a folyamatát.

# Irodalom áttekintés

# Célkitűzések

# Elméleti megalapozás

Ez a fejezet alapvető fogalmakat, folyamatokat és algoritmusokat dolgoz fel, amelyek elengedhetetlenek voltak a végső szoftver kifejlesztése közben.

## A relációs adatmodell alapvető fogalmai

*Relációs séma* egy rendezett pár , ahol attribútumhalmazt, pedig kényszerhalmazt jelöl (Mogin & Luković, dátum nélk.). Relációs séma megjelenési formája a reláció (ang. *relation*), amely korlátolt számú *sort* () (ang. *tuple*) tartalmaz (Mogin & Luković, dátum nélk.).

*Relációs adatbázis séma* egy rendezett pár , ahol relációs séma halmazt, pedig relációközi kényszerhalmazt jelöl (Mogin & Luković, dátum nélk.). Relációs adatbázis séma megjelenési formája a *relációs adatbázis* (ang. *relational database*) (Mogin & Luković, dátum nélk.).

*Funkcionális függőségek* (röviden függőségek) (ang. *functional dependency*) a relációs sémák integritását őrzik, más szóval a relációkban tárolt adatok közt vezetnek be összefüggéseket. Ha egy adott relációban egy funkcionális függőséget szemlélünk, akkor jelöli a baloldali-, míg a jobboldali attribútumhalmazt, melyek között funkcionális függőség van. Ebből kifolyólag a szóban forgó reláció bármely két sorára ( és ) érvényes a (4.1) képlet (Mogin & Luković, dátum nélk.). A függőségek megjelölésénél általában elhanyagoljuk a függőség megnevezését, így helyett csak írunk.

|  |  |  |
| --- | --- | --- |
|  |  | (4.1) |

Egy funkcionális függőség akkor *triviális*, ha érvényes .

Amennyiben egy relációban érvényes egy bizonyos függőséghalmaz, és egy szemlélt függőség az *igaz funkcionális függősége*, akkor az adott reláción érvényes az függőség is (Gajdos, 2019). Ezt következőképp jelöljük: . Másik meghatározás szerint ez azt jelenti, hogy logikai következménye az függőséghalmaznak (Mogin & Luković, dátum nélk.).

Bármelyik függőséghalmazon elvégezhetjük a relációs algebra *projekció* (ang. *projection*) műveletét. A (4.2) képlet mutatja be az függőséghalmaz projekcióját az attribútumhalmazra.

|  |  |  |
| --- | --- | --- |
|  |  | (.) |

Megjegyzés: az attribútumhalmazok úniójának (pl. ) helyett a rövidített megjelölést () használjuk a továbbiakban.

Azt a halmazt (4.3), amely az függőséghalmaz összes igaz függőségét (logikai következményét) tartalmazza, az halmaz *lezártjának* (ang. *closure*) hívják (Mogin & Luković, dátum nélk.).

|  |  |  |
| --- | --- | --- |
|  |  | (4.3) |

Két függéshalmaz ekvivalens, amennyiben érvényes a (4.4) képlet.

|  |  |  |
| --- | --- | --- |
|  |  | (.) |

Egy tetszőleges attribútumhalmaz lezártja az függéshalmazra való tekintettel a (4.5) képlettel van definiálva.

|  |  |  |
| --- | --- | --- |
|  |  | (.) |

Az attribútumhalmaz lezártjának a kiszámolásához két lépést használunk:

Az megjelölést a fenti lépéseknél az -edik ciklust jelöli. A 2. lépést mindaddig kell ismételni, amíg az és halmazok különböznek.

Az függőség *részleges függőség*, ha érvényes .

Az függőség *tranzítiv függőség*, ha érvényes .

Az attribútumhalmaz a relációs séma *kulcsa* (ang. *key*), amennyiben érvényes:

Relációs séma kulcsainak a kiszámolásához definiálnunk kell egy redukció műveletet (4.6), Ez a művelet egy attribútumhalmaz () minimális attribútumhalmazát határozza meg, amely nem tartalmaz felesleges attribútumokat (egy előre megadott függéshalmazra tekintettel).

|  |  |  |
| --- | --- | --- |
|  |  | (4.6) |

A relációs séma kulcsainak a kiszámolásához használt algoritmus:

*Elsődleges attribútumnak* (ang. *primary attribute*) nevezünk minden olyan attribútumot, amely a relációs séma kulcsát alkotja (4.6). *Másodlagos attribútumnak* (ang. *non-primary attribute*) nevezünk minden attribútumot, amely nem alkotja a relációs séma egyik kulcsát sem.

|  |  |  |
| --- | --- | --- |
|  |  | (.) |

## Normálformák

A *normálformák* (ang. *normal forms*) megszorítások a relációs séma tulajdonságaira vonatkozóan annak érdekében, hogy a sémákra illeszkedő relációkkal végzett műveletek során egyes nemkívánatos jelenségeket elkerülhessünk (Gajdos, 2019). Ezeket a nemkívánatos jelenségeket anomáliáknak hívják és beszúrási, módosítási vagy törlési műveletek során bukkanhatnak fel. Káros hatásuk akár az adott műveletek ellehetetlenítését is jelentheti. Ebben a dolgozatban a következő normálformákat mutatjuk be: , , és .

### 1NF

Az relációs séma kielégíti az (első normálforma) feltételét, amennyiben az halmazban kizárólag atomi értékeket hordozó attribútumok szerepelnek (Mogin, et al., 2004). Ez azt jelenti, hogy egy attribútum értéke sem lehet tömb vagy halmaz alakú. Az normálforma előfeltétele az összes többi normálformának, és ezt a tényt nem fogjuk külön kiemelni minden egyes normálformánál.

### 2NF

Az relációs séma kielégíti a (második normálforma) feltételét, amennyiben minden másodlagos attribútum teljesen függ a relációs séma összes kulcsától (4.7) (Mogin, et al., 2004).

|  |  |  |
| --- | --- | --- |
|  |  | (4.8) |

### 3NF

Az relációs séma kielégíti a (harmadik normálforma) feltételét, amennyiben minden másodlagos attribútum nem tranzitív függőségben van a relációs séma összes kulcsával (4.8) (Mogin, et al., 2004).

|  |  |  |
| --- | --- | --- |
|  |  | (4.9) |

Ahogy (Mogin, et al., 2004) kiemeli, létezik egy alternatív definíciója is a normálformának. E definíció szerint minden nem triviális függőség bal oldalának tartalmaznia kell a relációs séma egy kulcsát, amennyiben a jobb oldali attribútumhalmaz tartalmaz másodlagos attribútumot (4.9) (Mogin, et al., 2004).

|  |  |  |
| --- | --- | --- |
|  |  | (4.10) |

### BCNF

Az relációs séma kielégíti a (ang. *Boyce[[1]](#footnote-1)-Codd[[2]](#footnote-2) normal form*), amennyiben minden nem triviális függőség bal oldala tartalmazza a relációs séma egy kulcsát (4.10) (Mogin, et al., 2004).

|  |  |  |
| --- | --- | --- |
|  |  | (4.11) |

## Normalizációs algoritmusok

Az adatbázisok normalizálásának célja, hogy egy (vagy több) relációs sémát egy bizonyos normálformára vezessen, a nemkívánatos anomáliák elkerülése érdekében.

### Szintézis

A szintézis algoritmus kiindulópontja az ún. univerzális relációs séma , ahol az az univerzális attribútumhalmazt, az pedig az univerzális függéshalmaz jelöli. A szintézis elvégeztével darab relációs sémát kapunk, melyek kielégítik a normálformát. A relációközi megszorításokkal kiegészülve megkapjuk a szintézis kimenetét, vagyis az adatbázis sémát (4.1).

|  |  |  |
| --- | --- | --- |
|  |  | (.) |

A szintézis algoritmusa:

1. Minimális függéshalmaz meghatározása

Fontos megjegyezni, hogy a minimális függéshalmazban () ekvivalens a kiinduló függéshalmazzal (4.2). A minimális függéshalmazban a függőségek jobboldali attribútumhalmazában csak egyetlen attribútum található (4.3), a függőségek teljes függőségek (4.4), valamint nincs olyan függőség, amelyik elhagyható (4.5) (Gajdos, 2019).

|  |  |  |
| --- | --- | --- |
|  |  | (.) |
|  |  | (4.14) |
|  |  | (4.15) |
|  |  | (4.16) |

A minimális függéshalmaz fent említett tulajdonságait a következő algoritmussal érhetjük el:

for X->Y ∈ F:

Fmin <- Fmin + {X->A | A ∈ Y}

for X->A ∈ Fmin:

for B ∈ A:

if Fmin |= X\{B}->A:

Fmin <- Fmin\{X->A} + {X\{B}->A}

for X->A ∈ Fmin:

if X->A ∈ (Fmin\{X->A})+:

Fmin <- F\{X->A}

1. Minimális függéshalmaz átalakítása

A minimális függéshalmazt fel kell osztani partícióhalmazra (4.6), ahol minden egyes partícióba olyan függőségeket csoportosítunk, melyeknek a baloldali halmazai megegyeznek (4.7).

|  |  |  |
| --- | --- | --- |
|  |  | (.) |
|  |  | (4.18) |

A megformált partíciókat egyesíteni kell az ekvivalens baloldali halmazok szerint, vagyis minden pár partíció és , amelyre érvényes , azokat egyesítjük partícióvá, majd a partícióhalmazt módosítjuk (4.8).

|  |  |  |
| --- | --- | --- |
|  |  | (.) |

Miután ezeket a bizonyos partíciókat egyesítettük, felmerülhet az a veszély, hogy tranzitív függőségeket hozunk létre ezekben a partíciókban, ezért létre kell hozni egy halmazt, mely tartalmazza a tranzitivitás megelőzésére alkalmas függőségeket (4.9). Annak érdekében törekedünk a tranzitív függőségek felszámolására, hogy a relációs sémák teljesíteni tudják majd a normálformát. Ezeknek az újabb függőségeknek a hozzáadásával lehet, hogy sikerült kiküszöbölni a tranzitív függőségeket, de potenciálisan elhagyható függőségek jöttek létre. Az érintett partíciókból átmenetileg ki kell vonni ezeket a függőségeket (4.10), majd törölni kell a feleslegessé váltakat.

|  |  |  |
| --- | --- | --- |
|  |  | (.) |
|  |  | (4.21) |

Az elhagyható függőségek törlése végett létrehozunk egy halmazt (4.11), majd ennek a halmaznak a tekintetében végezzük a függőségek egyszerűsítését – a(z) (4.5) képlethez hasonlóan.

|  |  |  |
| --- | --- | --- |
|  |  | (.) |

Miután a megfelelő partíciókból törlésre kerültek a felesleges függőségek, visszaállítjuk a halmazbeli függőségeket a megfelelő partíciókba.

1. Relációs adatbázis séma létrehozása

Minden partíció egy relációs sémát alkot, ahol az halmaz jelenti a séma kulcsát. A partíció függőségeiben előforduló attribútumok pedig a séma attribútumhalmazát képezik. A relációközi megszorításokat az idegen kulcsok megszorításai alkotja.

1. Veszteségmentes sémafelbontás megőrzése

Annak érdekében, hogy meggyőződjünk a veszteségmentes sémafelbontásról, le kell ellenőrizni, hogy bár egy relációs séma kulcsa megegyezik az univerzális relációs séma kulcsával. Ha igen, akkor veszteségmentesen bontottuk fel a sémát. Amennyiben a válasz nem, további relációs sémára lesz szükségünk, melynek kulcsa megegyezik az univerzális relációs séma egyik szabadon választott kulcsával, az attribútumhalmaz pedig a kiválasztott kulcsot képező attribútumokkal.

### Dekompozíció

A dekompozíció algoritmus kiindulópontja az ún. univerzális relációs séma , ahol az az univerzális attribútumhalmazt, az pedig az univerzális megkötéshalmazt jelöli. A halmaz magában foglalja az univerzális relációs séma függőségeit valamint a többértékű függőségeket is (ang. *multivalued dependency*). Mivel a dolgozat csak olyan algoritmusokat taglal, amelyek legfeljebb a normálformát elégítik ki, ezért nem fogjuk figyelembe venni a többértékű függőségeket. A dekompozíció elvégeztével darab relációs sémát kapunk, melyek kielégítik a normálformát. A relációközi megszorításokkal kiegészülve megkapjuk a dekompozíció kimenetét, vagyis az adatbázis sémát.

A dekompozíció algoritmusa:

1. Megfelelő függőség kiválasztása, amely a dekompozíció lépéseinek a tárgyát képezik

A dekompozíció lépéseinek az első eleme a megfelelő függőség kiválasztása, ami alapján felosszuk az adott relációs sémát. A kívánt függőséget három kritérium alapján tudjuk kiválasztani:

* 1. P1 kritérium (4.12) szerint egy nemtriviális függőséget kell választanunk, ahol az halmaz nem szuperkulcs, valamint a függőséghalmaz meghatározott szétválasztása nem jár függőségvesztéssel.

|  |  |  |
| --- | --- | --- |
|  |  | (.) |

* 1. P2 kritérium (4.13) szintén nemtriviális függőség kiválasztását terjeszti elő, melynek jobb- és baloldali attribútumhalmazainak az úniója különböznek az adott sémareláció attribútumhalmazától (), valamint a függőséghalmaz meghatározott szétválasztása nem jár függőségvesztéssel.

|  |  |  |
| --- | --- | --- |
|  |  | (.) |

* 1. P3 kritérium (4.14) szintén nemtriviális függőség kiválasztását terjeszti elő, ahol az halmaz nem szuperkulcs. Az előző két kritériummal ellentétben a P3 kritérium nem szabja feltételként a függőségvesztés kitételt.

|  |  |  |
| --- | --- | --- |
|  |  | (.) |

A megfelelő függőség kiválasztásánál ügyelni kell arra, hogy minél magasabb kritérium teljesüljön.

1. Relációs séma szétválasztása a kiválasztott függőség alapján

Amennyiben sikerült kiválasztani a megfelelő függőséget, akkor az adott relációs sémát a következő két relációs sémára tudjuk felbontani:

Ilyen felbontás mellett teljesülnek a veszteségmentes összevonás feltételei (4.15), ahol a és halmazok a megfelelő relációs sémák kulcshalmazait jelölik.

|  |  |  |
| --- | --- | --- |
|  |  | (.) |

1. Normálforma vizsgálat

A szétbontott relációs sémákat normálforma vizsgálat alá helyezzük, és amennyiben nem elégítik ki a normálformát, további dekompozíciónak vetjük alá a sémákat, kezdve az algoritmusban szereplő 1) ponttal. Ezt a folyamatot rekurzív módon hajtjuk végre, vagyis az első pontban szereplő séma helyét a második pontban kapott sémák veszik át (amennyiben azok nem teljesítik a normálformát). Azokat a relációs sémákat, amelyek teljesítik a normálformát, elvégzetteknek (dekomponáltaknak) tekintünk.

1. A dekompozíciós fa kiértékelése

A relációs sémák szétválasztása során bináris fa jön létre, melynek levelei alkotják a dekomponált relációs sémahalmazt. Ezt a sémahalmazt további kiértékelésnek vetjük alá, mégpedig az ekvivalens kulccsal rendelkező sémákat összevonjuk. Ezzel a lépéssel visszanyerhetünk időközben elvesztett függőségeket – amennyiben a P3 kritérium (4.14) alapján tudtunk csak függőséget választani a dekompozíció során. Ezekkel a visszanyert függőségekkel viszont kockáztatjuk az elért normálformát, de a függőségek megőrzése érdekében beáldozhatjuk ezeket a sémákat.

1. Veszteségmentes sémafelbontás megőrzése

A szintézis algoritmusához hasonlóan meg kell győződjünk a veszteségmentes sémafelbontásról, amit úgy érünk el, hogy leellenőrizzük, hogy bár egy relációs séma kulcsa megegyezik az univerzális relációs séma kulcsával.

## Szoftvermodellezési szempontok

Szoftvermodellezéshez a UML (ang. *Unified Modeling Language*) modellnyelvet használtam, hogy egy konkrét programnyelvtől független leírást mutathassak be a fejlesztett szoftverről. Mivel a szoftver tervezésétől kezdve objektum-orientált programnyelvi paradigmában gondolkodtam, ezért adatmodellezéshez osztálydiagramot használtam. Az adatbázis normalizálási algoritmusokat szekvenciadiagramokkal terveztem ki.

A következő fejezet részletesen bemutatja a szoftverfejlesztéshez használt diagramokat, valamint konkrét kódrészletek is magában foglal.

# Gyakorlati megvalósítás

# Eredmények

# Tárgyalás

# Összefoglalás

# Irodalomjegyzék

Bahmani, A., Naghibzadeh, M. & Bahmani, B., 2008. *Automatic database normalization and primary key generation.* Niagara Falls, 2008 Canadian Conference on Electrical and Computer Engineering.

Čeliković, M. és mtsai., 2021. *Adatbázisok 2 - labor gyakorlatok gyűjtemény,* Újvidék: Műszaki Tudományok Kara, Újvidéki Egyetem.

Diederich, J. & Milton, J., 1988. New methods and fast algorithms for database normalization. *ACM Transactions on Database Systems,* 13(3), pp. 339-365.

Du, H. & Wery, L., 1999. Micro: A normalization tool for relational database designers. *Journal of Network and Computer Applications,* 22(4), pp. 215-232.

Gajdos, S., 2019. *Adatbázisok.* Budapest: Műegyetemi Kiadó.

Georgiev, N., 2008. *A Web-Based Environment for Learning Normalization of Relational Database Schemata,* Umeå: Umeå University.

Kordić, S. és mtsai., 2018. *Baze podataka - zbirka zadataka.* Újvidék: Műszaki Tudományok Kara, Újvidéki Egyetem.

Loukides, M., 2022. *Technology Trends for 2022,* Amerikai Egyesült Államok: O'Reilly Media, Inc..

Mitrovic, A., 2002. *NORMIT: a Web-enabled tutor for database normalization.* Auckland, International Conference on Computers in Education.

Mogin, P., Luković, I. & Govedarica, M., 2004. *Principi projektovanja baza podataka,* Újvidék: Műszaki Tudományok Kara, Újvidéki Egyetem.

Mogin, P. & Luković, I., dátum nélk. *Principi baza podataka.* Újvidék: Műszaki Tudományok Kara, Újvidéki Egyetem.

Műszaki Tudományok Kara, Újvidék, 2021. *Adatbázisok 2.* [Online]   
Available at: http://ftn.uns.ac.rs/571482529/databases-2  
[Hozzáférés dátuma: 2022. március 16.].

Ramel, D., 2015. *Relational Databases Still Reign in Enterprises, Survey Says.* [Online]   
Available at: https://esj.com/articles/2015/04/23/database-survey.aspx  
[Hozzáférés dátuma: 2022. március 30.].

ScaleGrid.io, 2019. *2019 Database Trends.* [Online]   
Available at: https://scalegrid.io/blog/2019-database-trends-sql-vs-nosql-top-databases-single-vs-multiple-database-use/  
[Hozzáférés dátuma: 2022. március 30.].

1. Raymond F. Boyce (1946–1974) amerikai informatikus [↑](#footnote-ref-1)
2. Edgar F. Codd (1923–2003) angol informatikus [↑](#footnote-ref-2)