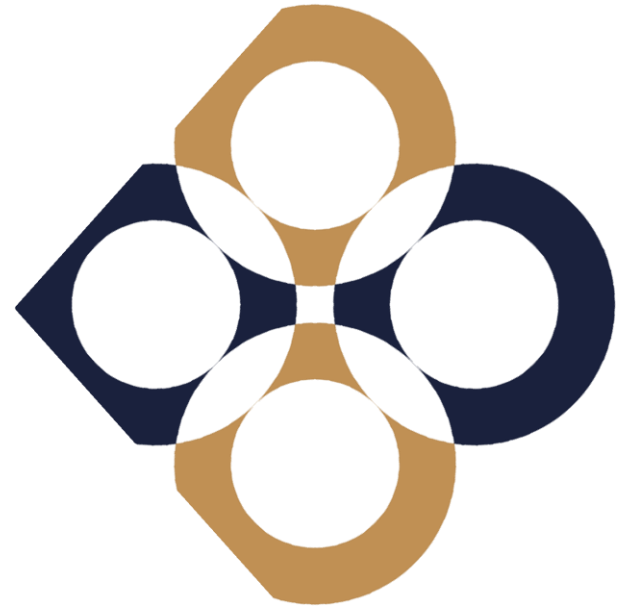


Adatbázisok előadás I.

Alapfogalmak



Miről lesz szó?

1. Tudnivalók
2. Miről fogunk tanulni?
3. Bevezetés
4. Alapfogalmak

1. Tudnivalók

Követelmények – részletesebb leírás a tantárgyi útmutatóban

- Félév közben
 - Részvétel a gyakorlatokon (Maximálisan 25% hiányzás megengedett)
 - ZH-k és tesztek megírása, feladatok határidőre történő leadása

A számonkérések tervezett üteme*

- ☐ ZH1 – negyedik gyakorlat elején (SQL alapok + elméleti teszt)
- ☐ ZH2 – hetedik gyakorlat elején (haladó SQL + elméleti teszt)
- ☐ ZH3 - tizedik gyakorlat elején (dokumentum és gráf adatbázisok + elméleti teszt)
- ☐ Félévzáró ZH – tizenkettedik gyakorlaton (elmélet + gyakorlatból minden)

- ☐ Házi feladatok: 2 alkalommal, egy hetes határidővel (kiadás: 4. és 8. gyakorlat)
- ☐ Beadandó feladat (esettanulmány): határidő a félév vége, kiadás: 6. gyakorlat)

* Az intenzív hét és a tavaszi szünet miatt maximum 12 gyakorlat lesz

- ☐ Félévközi tesztek és zh-k – 30%
- ☐ Házi feladatok – 10%
- ☐ Esettanulmány – 20%
- ☐ Félévzáró teszt és zh – 40%

Jeles: 90%-tól, Jó: 80%-tól, Közepes: 70%-tól, Elégséges: 60%-tól, Elégtelen: 60% alatt

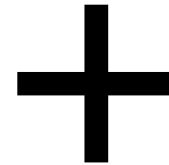
Kapcsolat

- Email – normál esetben
 - geza.molnar2@uni-corvinus.hu
 - A beazonosításhoz a hallgató adja meg a csoportját és NEPTUN-kódját is
- TEAMS – ha sürgős

2. Miről fogunk tanulni?

Tematika

- Alapfogalmak
- Relációs adatmodell
- Relációs algebra
- Adatbázisok tervezése
- Adattárházak, adattárházfejlesztés a gyakorlatban
- Nem relációs adatbázisok
- Adatbáziskezelő rendszerek a gyakorlatban



Gyakorlatok I-VII.: SQL-nyelv



Az előadástól független tananyag

Fókusz: lekérdezések (SELECT)

Eszközök:

- ☐ Azure Data Studio
- ☐ Microsoft Fabric
- ☐ SQL Server Express
- ☐ Online SQL Editor

Gyakorlatok VIII-XII.: NoSQL



Az előadással szinkronban

Fókusz: lekérdezések SQL-es szemszögből

Eszközök:

- ☐ MongoDB Atlas/Compass
- ☐ Neo4J Sandbox/Desktop
- ☐ Redis Cloud/CLI
- ☐ Cassandra Datastax Astra

Kapcsolódó tantárgyak/területek

Szoftverfejlesztés

Üzleti
intelligencia

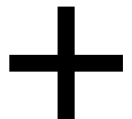
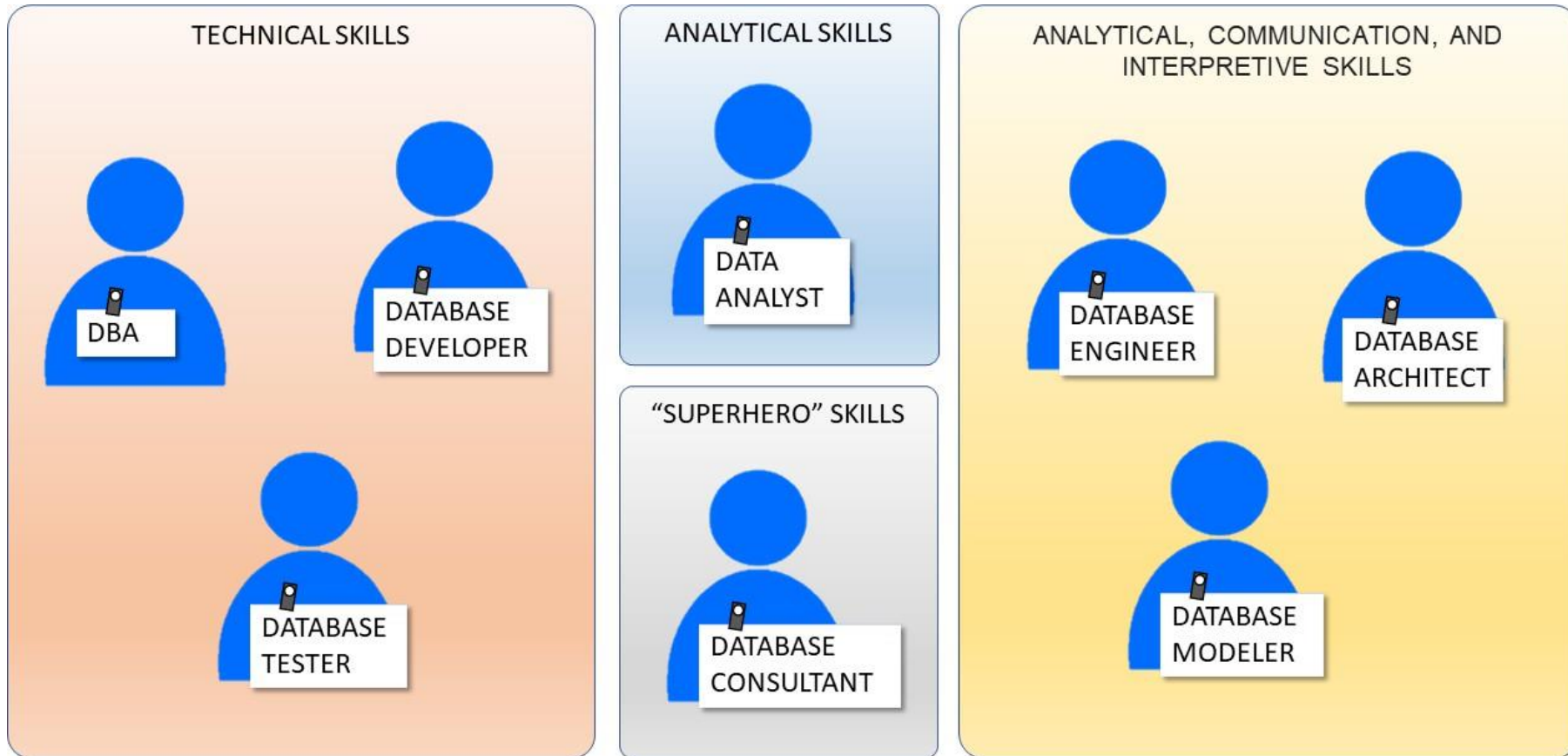
Adatelemzés,
adattudomány

Információs
rendszerek

Döntéstámogatás

IT Architektúrák

Hol alkalmazható a megszerzett tudás?



A kapcsolódó területek pozíciói, pl: webfejlesztő, statisztikus, gazdaságinformatikus stb.

3. Bevezetés

Miért van szükségünk adatbázisokra?



Nagy
adatmennyiség



Sok
felhasználó



Konziszten-
cia



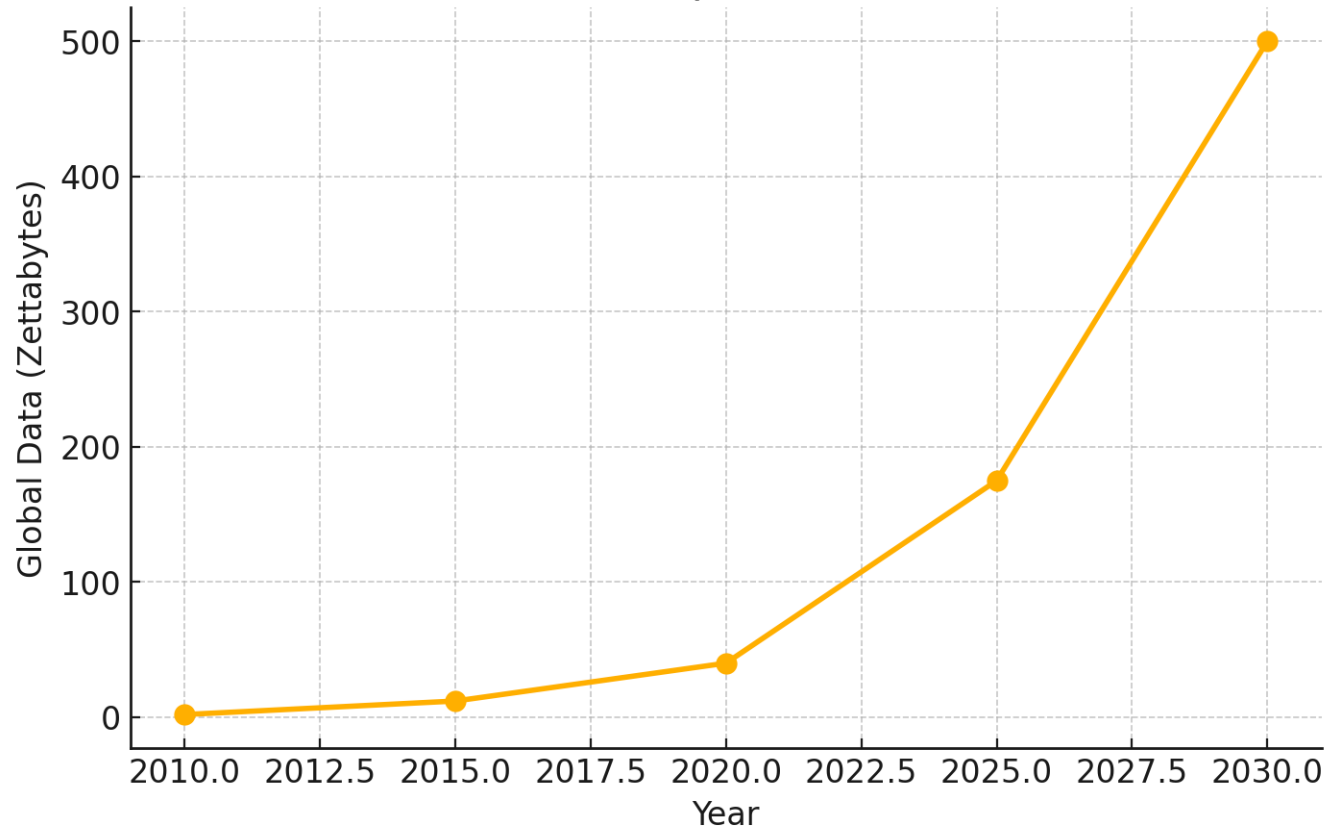
Adatok
elemzése

Fontos: az adatok lekérdezhetők legyenek!

Kahoot kérdés 01

Nagy adatmennyiség

Global Data Explosion Over Time



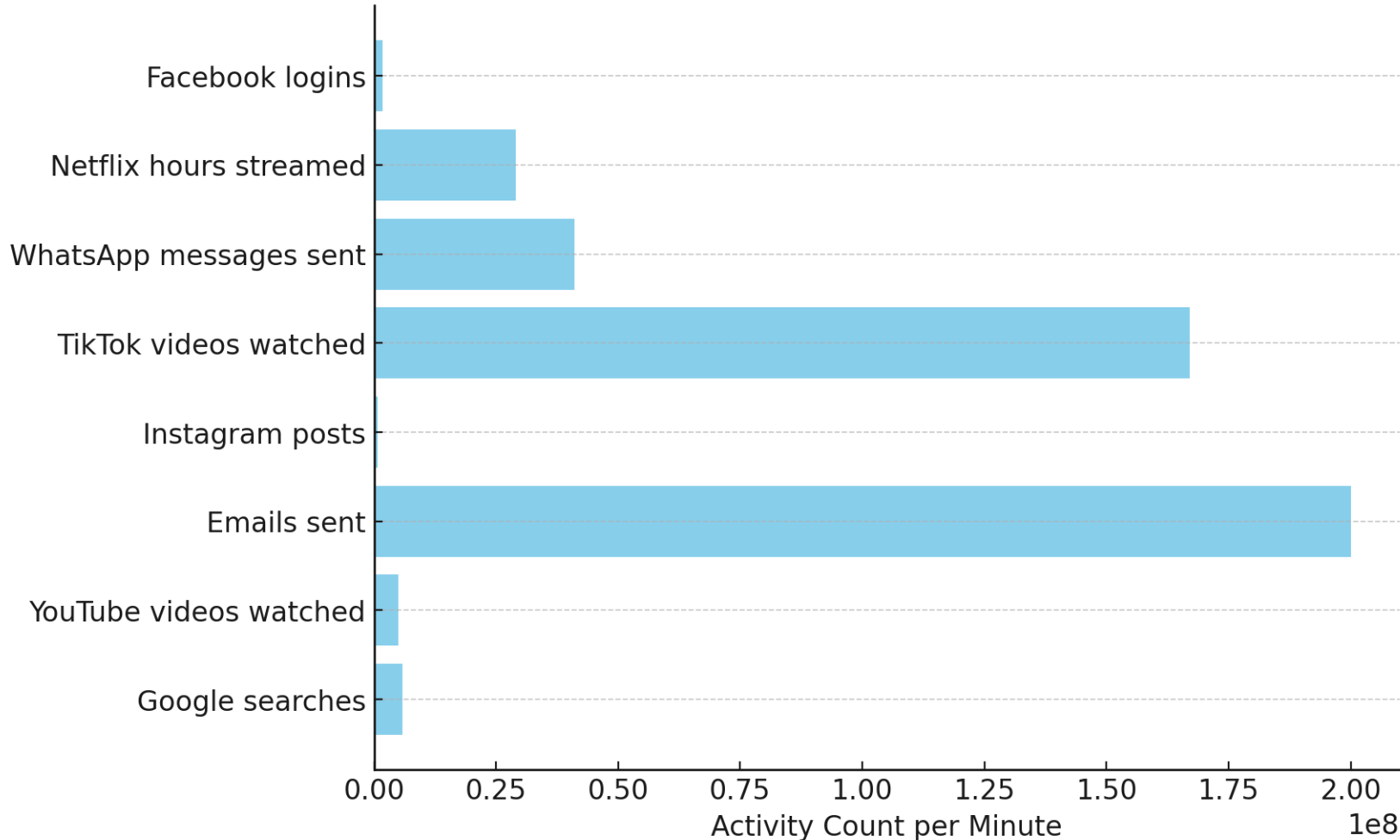
2020: 40 zettabyte (ZB)

2025 előrejelzés: 175 ZB

1 ZB = 1 milliárd terabyte = 1 billió gigabyte

Nagy adatmennyiség (folyt.)

What Happens Every Minute on the Internet?



Equivalent Units

DVD (4.7 GB)	1 089 362
USB Flash Drive (128 GB)	40 000
SSD (1 TB)	5 000
HDD (10 TB)	500
Data Center Server (100 TB)	50
Human Brain Memory (2.5 PB)	2 000

Konzisztencia

A **konzisztencia** azt jelenti, hogy az adatok mindig érvényes állapotban maradnak, megfelelnek az előírt szabályoknak és integritási feltételeknek.

Adatbázis nélkül (pl. papíralapú nyilvántartás) inkonzisztens adatok keletkezhetnek.

Példák

- ☐ **Banki Tranzakciók.** Ha egy pénzáttétel közben elakad az adatfrissítés, a pénz eltűnhet. Adatbázis esetén ez nem fordulhat elő.
- ☐ **Webshop Rendelések.** Ha a raktárkészlet nincs frissítve, egy vásárló fizethet egy már elfogyott termékért. Az adatbázis biztosítja, hogy a rendelési és készletadatok együtt frissüljenek.
- ☐ A **papíralapú nyilvántartás szétesik**, ha több ember egyszerre dolgozik rajta

A kezdetek (1960 előtt)

- Még nem voltak adatbázisok
- Az adatokat a felhasználói programok kezelték
 - Az adatokat egyszerű fájlokban (pl. szöveges fájl) tárolták
 - A tárolás sémája esetleges volt
 - Lekérdezni csak a programból lehetett
- Minden egyes adattárolási és lekérdezési feladatra külön programot kellett írni

1960-as évek eleje

Egyre
problémásabb
adatkezelés

Nem mindegy,
hogyan
tároljuk az
adatokat

Szükség van
valamilyen
logikára

Megjelentek
az első
adatmodellek

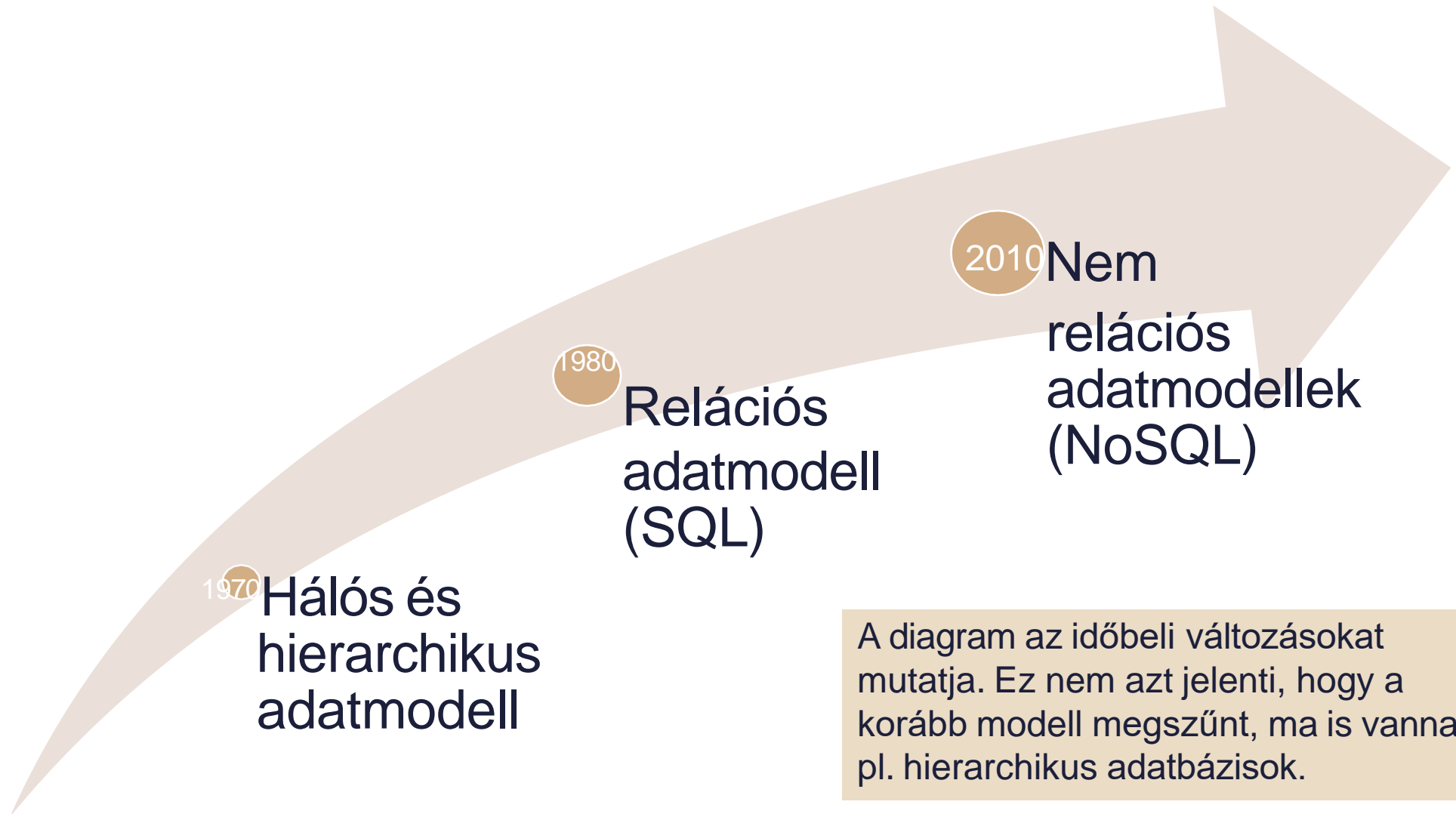
Az 1960-as évek

- Az első adatbázisok és adatmodellek megjelenése
 - **IDS** (Integrated Data Store)
 - Az egyik **legelső adatbáziskezelő**
 - **Charles Bachman** fejlesztette ki
 - **CODASYL**
 - A **hálós adatmodell** megfogalmazása
 - **COBOL** programozók fejlesztése
 - **IMS** (Information Management System)
 - **IBM** fejlesztése
 - **Hierarchikus adatmodellt** használt

Az 1960-as évek után

- Sorra jelentek meg az új adatmodellek, és ezekre épülve az új adatbáziskezelő rendszerek
- Minden adatmodellnek voltak közös elemei
 - Olyan dolgok, amelyekről adatokat tárolunk (egyedek)
 - Az adott dolgok jellemzői (tulajdonságok)
 - Az adott dolgok közötti összefüggések (kapcsolatok)
- **A domináns adatmodell a relációs lett**
- Napjainkban megjelentek és kezdenek elterjedni a nem relációs (NoSQL) adatbázisok

Az adatmodellek fejlődésének mérföldkövei



4. Alapfogalmak

Alapfogalmak

- Adatmodell
- Adatbázis
- Adatbáziskezelő rendszer
- Adatbázis rendszer

Egyedek (entitások)

A valós világ olyan dolgai, amelyek minden más dologtól megkülönböztethetők

Pl: autó, személy, termék

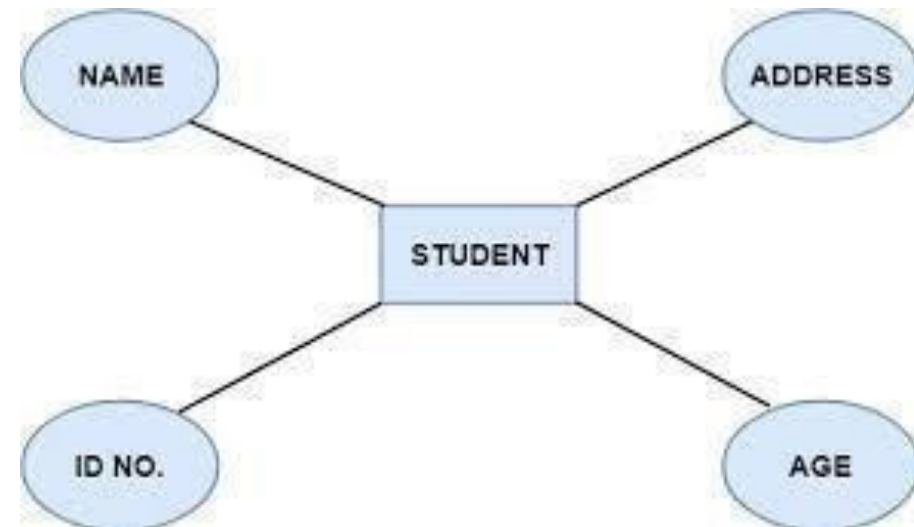
Tulajdonságok

Az egyedeket tulajdonságaikkal írjuk le, azaz a tulajdonságok az egyedek belső szerkezetét jelentik.

Pl:

A tanuló egyed tulajdonságai lehetnek:

- Azonosító
- Név
- Életkor
- Cím stb.

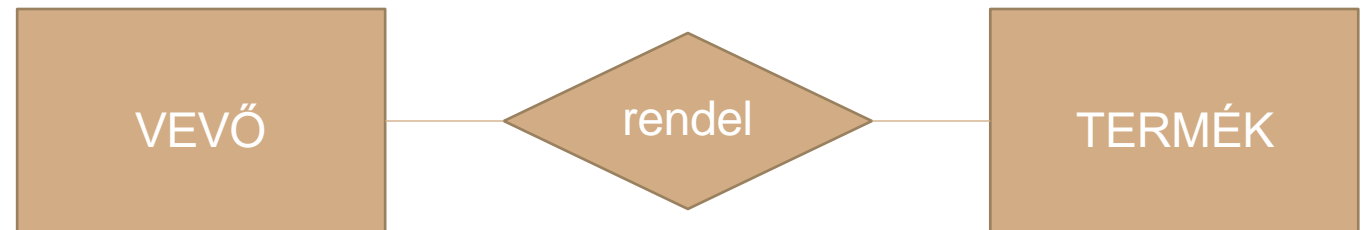


Kapcsolatok

Az egyedek közötti viszonyt kapcsolatnak nevezzük.
Másképpen fogalmazva a kapcsolat az egyed külső szerkezete

Példák:

VEVŐ-TERMÉK
TERMÉK-ELADÁS
TANULÓ-ISKOLA



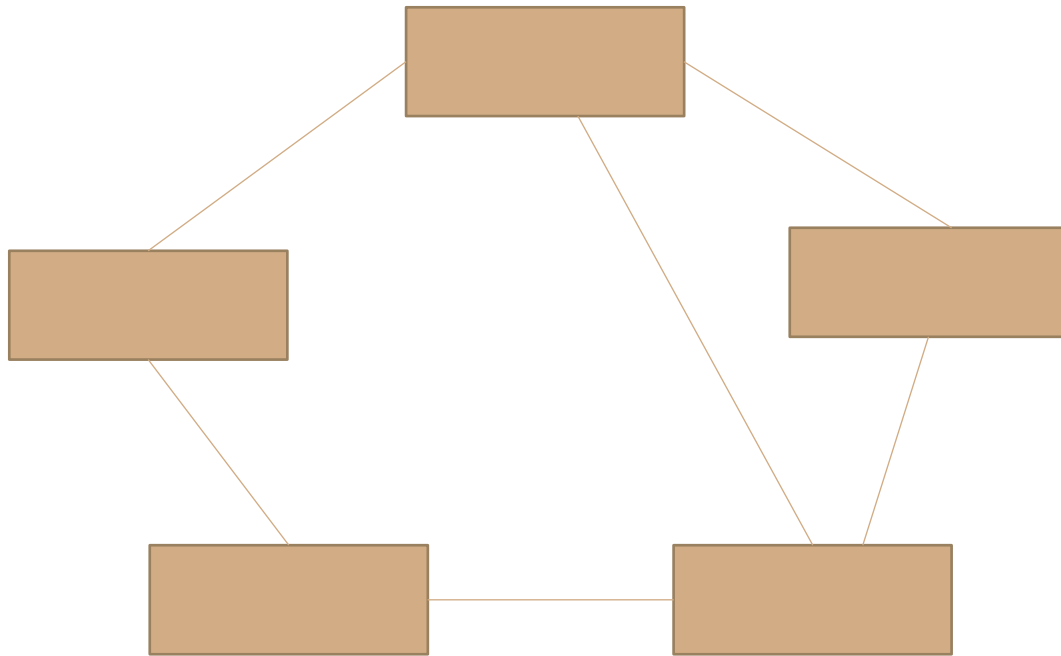
Alapfogalmak - Adatmodell

Adatmodellnek nevezzük az egyedek, tulajdonságok és kapcsolatok halmazát

Tipikus adatmodellek:

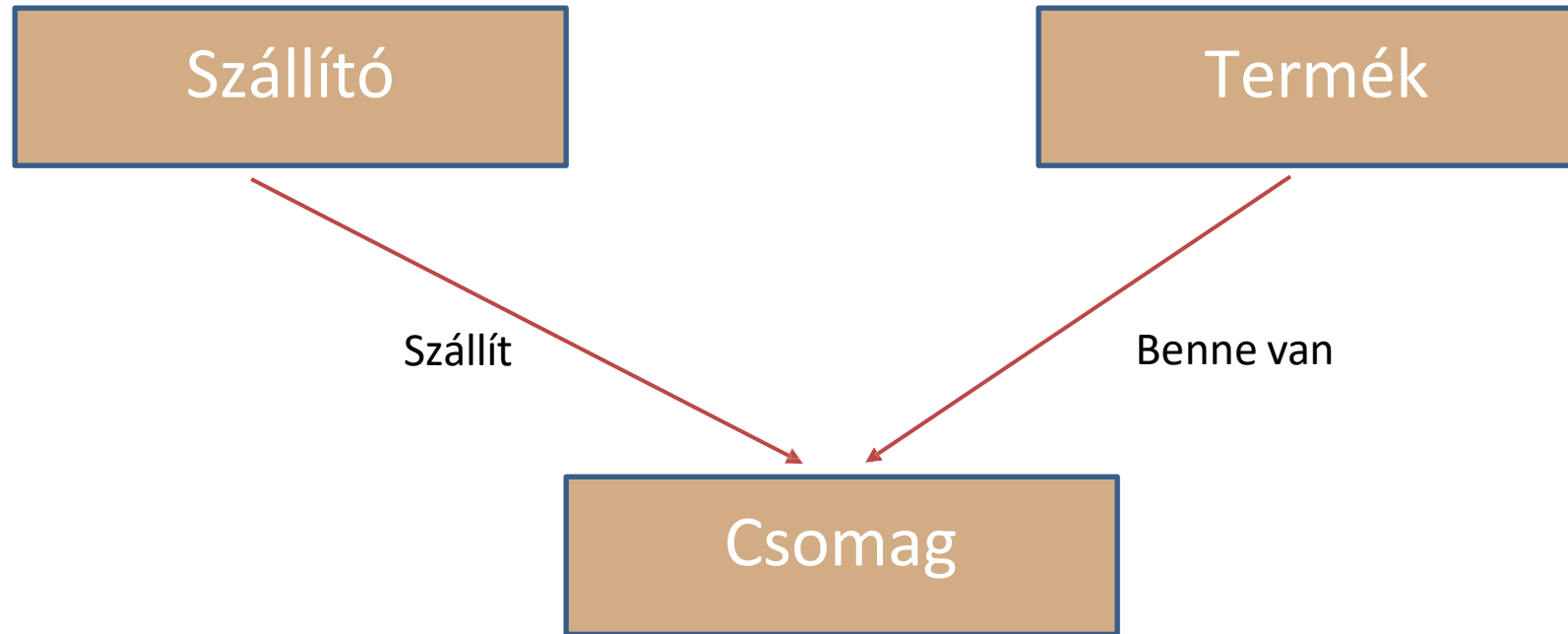
- ☐ Hálós adatmodell (gráf)
- ☐ Hierarchikus adatmodell (fa)
- ☐ **Relációs adatmodell (táblák)**
- ☐ Nem relációs adatmodellek

Hálós adatmodell



Olyan gráf, ahol a csomópontok az egyedek, az élek a kapcsolatok

Hálós adatmodell – példa (séma)



Hálós adatmodell – példa (adatokkal)

Szállító

Kod	Név	Cím
001	XY Kft	BP
002	ZZ Rt	Vác

Termék

Kód	Név	EgysÁr
T01	Tej	250
T02	Tea	500
T03	Kóla	350

Szülő	Gyerek

Csomag

DB	Ár
25	50000
33	16500

Szülő	Gyerek

Hálós adatmodell – előnyök és hátrányok



Egyszerű elv

Többféle
kapcsolat
kezelése

Az adatok
könnyen
elérhetők

Adatintegritás,
adatfüggetlenség



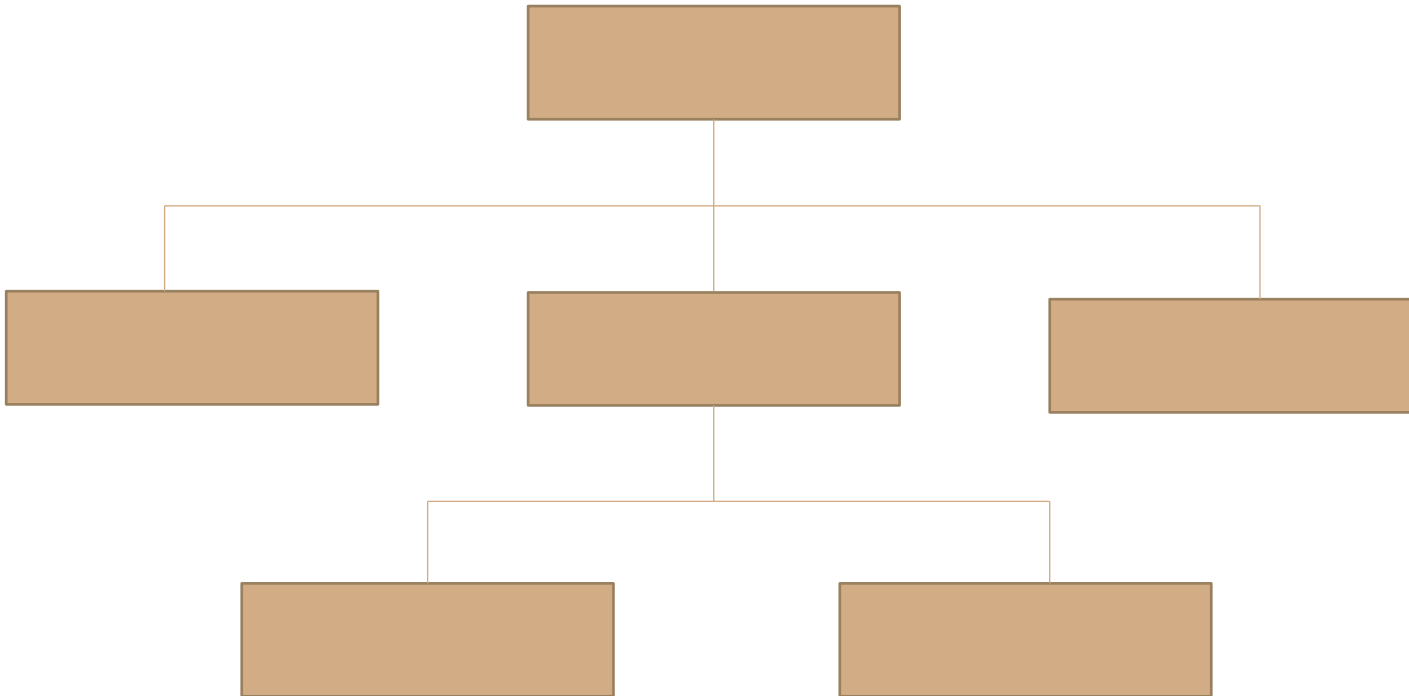
A mutatók
kezelése

Komplex
lekérdezések

Adatok
törlése,
módosítása

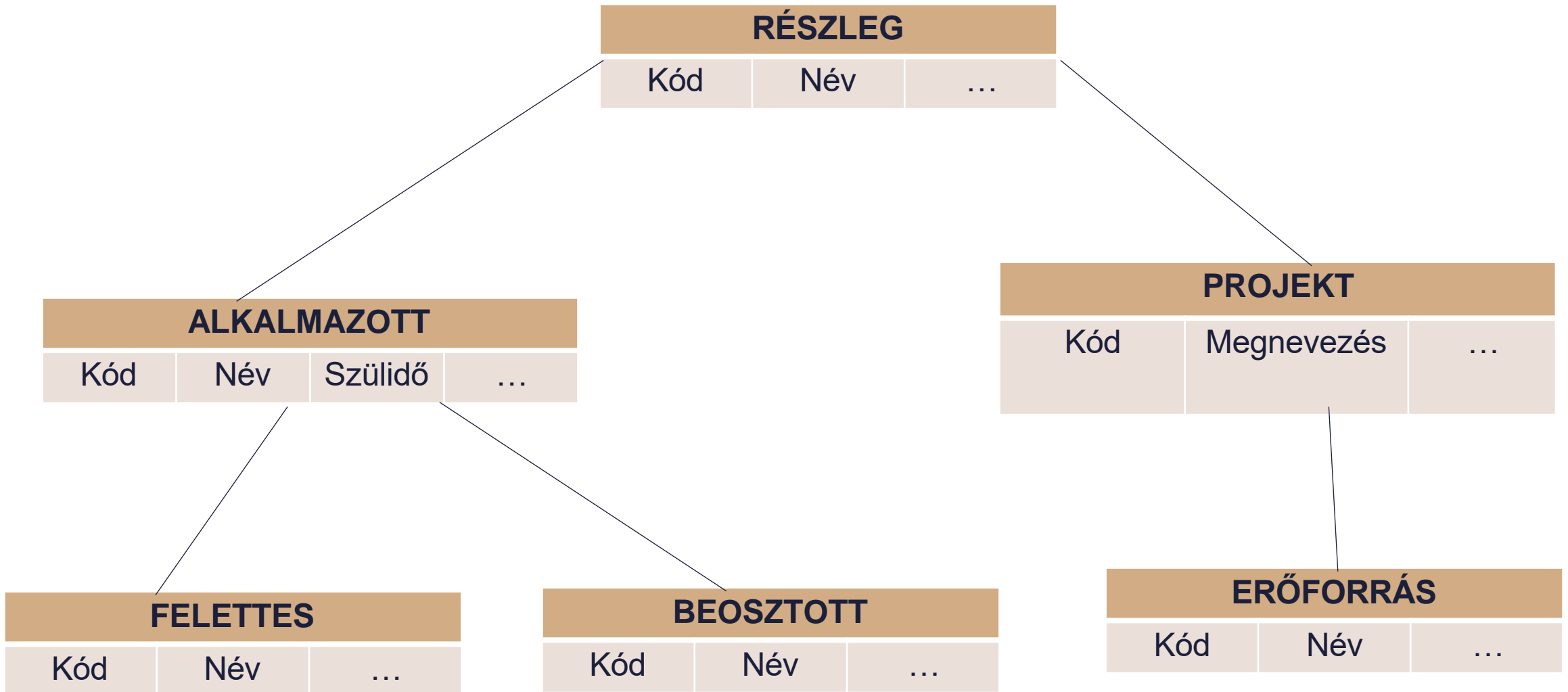
Séma
módosítása
nehéz

Hierarchikus adatmodell

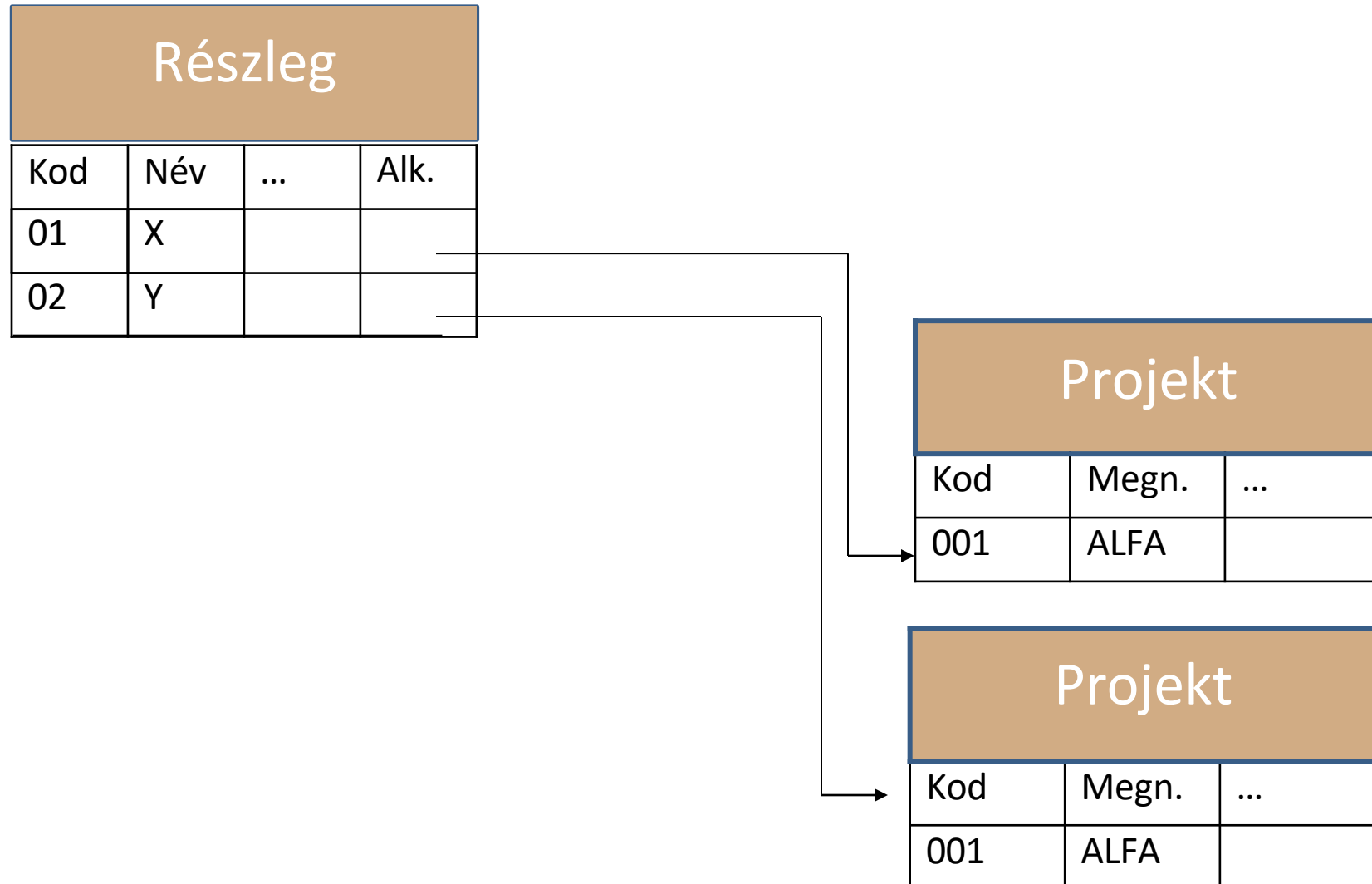


Olyan fa struktúra,
ahol a csomópontok
az egyedek, az élek
a kapcsolatok

Hierarchikus adatmodell – példa (séma)



Hierarchikus adatmodell – példa (adatokkal) - részlet



Hierarchikus adatmodell – előnyök és hátrányok



Egyszerű elv

Adatok
megosztása

Az adatok
gyorsan
elérhetők

Természetes
hierarchiák
megvalósítása



Duplikációk

A fizikai
adattárolás
programfüggő

Adatok
törlése,
módosítása

Séma
módosítása
nehéz

Edgar Codd (~1970)

- Felmérte a hálós és hierarchikus modellek hátrányait
- Új alapelveket javasolt ([12 pontban](#))
- Ötleteinek lényege
 - Egyszerű adatszerkezetek használata
 - Adatok elérése magasszintű programnyelven
 - Fizikai adattárolás függetlensége

➔ Megalkotta a relációs adatmodellt

Edgar Codd cikke (1970)

Information Retrieval

P. BAXENDALE, Editor

A Relational Model of Data for Large Shared Data Banks

E. F. CODD

IBM Research Laboratory, San Jose, California

Future users of large data banks must be protected from having to know how the data is organized in the machine (the internal representation). A prompting service which supplies such information is not a satisfactory solution. Activities of users at terminals and most application programs should remain unaffected when the internal representation of data is changed and even when some aspects of the external representation are changed. Changes in data representation will often be needed as a result of changes in query, update, and report traffic and natural growth in the types of stored information.

Existing noninferential, formatted data systems provide users with tree-structured files or slightly more general network models of the data. In Section 1, inadequacies of these models are discussed. A model based on n -ary relations, a normal form for data base relations, and the concept of a universal data sublanguage are introduced. In Section 2, certain operations on relations (other than logical inference) are discussed and applied to the problems of redundancy and consistency in the user's model.

KEY WORDS AND PHRASES: data bank, data base, data structure, data organization, hierarchies of data, networks of data, relations, derivability, redundancy, consistency, composition, join, retrieval language, predicate calculus, security, data integrity

CR CATEGORIES: 3.70, 3.73, 3.75, 4.20, 4.22, 4.29

The relational view (or model) of data described in Section 1 appears to be superior in several respects to the graph or network model [3, 4] presently in vogue for non-inferential systems. It provides a means of describing data with its natural structure only—that is, without superimposing any additional structure for machine representation purposes. Accordingly, it provides a basis for a high level data language which will yield maximal independence between programs on the one hand and machine representation and organization of data on the other.

A further advantage of the relational view is that it forms a sound basis for treating derivability, redundancy, and consistency of relations—these are discussed in Section 2. The network model, on the other hand, has spawned a number of confusions, not the least of which is mistaking the derivation of connections for the derivation of relations (see remarks in Section 2 on the “connection trap”).

Finally, the relational view permits a clearer evaluation of the scope and logical limitations of present formatted data systems, and also the relative merits (from a logical standpoint) of competing representations of data within a single system. Examples of this clearer perspective are cited in various parts of this paper. Implementations of systems to support the relational model are not discussed.

1.2. DATA DEPENDENCIES IN PRESENT SYSTEMS

The provision of data description tables in recently developed information systems represents a major advance toward the goal of data independence [5, 6, 7]. Such tables facilitate changing certain characteristics of the data representation stored in a data bank. However, the variety of data representation characteristics which can be changed *without logically impairing some application programs* is still quite limited. Further, the model of data with which users interact is still cluttered with representational properties, particularly in regard to the representation of col-

Kahoot kérdés 02

Relációs adatmodell


Relációs adatmodell esetén

- Az egyedek táblák (relációk)
- A tulajdonságok a táblák oszlopai (attribútumok)
- A kapcsolatok indirekt formában vannak jelen

Relációs adatmodell - példa

student_id	name	age
1	Akon	17
2	Bkon	18
3	Ckon	17
4	Dkon	18

subject_id	name	teacher
1	Java	Mr. J
2	C++	Miss C
3	C#	Mr. C Hash
4	Php	Mr. P H P



student_id	subject_id	marks
1	1	98
1	2	78
2	1	76
3	2	88

Mi történt 1980 és 2010 között?

- A relációs adatmodell egyeduralkodó lett
 - Megjelentek újabb adatmodellek (pl. objektumorientált), de tömegesen nem terjedtek el
 - Elterjedtek viszont az adattárházak
 - Tömegessé váltak az internetes, adatbázisokra épülő alkalmazások
 - A kezelendő adatmennyiség sokszorosára nőtt
- ➔ A relációs adatbázisok egyre több kihívással néztek szembe

2010 után

Nem relációs (NoSQL) adatmodellek megjelenése

- A legtöbb esetben egy régi ötlet újragondolását jelentik
- Nincs szabványos lekérdező nyelvük
- Részletesebben ld. 9-13. előadásokon

Napjainkban + Közeljövőben

Relációs és nem relációs adatbázisok együttélése

- A közöttük lévő határok elmosódnak (NewSQL)
 - Pl: CockroachDB, TiDB, Google Spanner
- Hibrid rendszerek
- Többmodelles adatbázis rendszerek
 - Pl: ArangoDB, CosmosDB, OrientDB

Napjainkban + Közeljövőben (folyt.)

- Felhős adatbázis rendszerek (DBaaS)
 - Adatbázisok: Google BigQuery, Azure SQL Database, AWS RDS
 - Serverless megoldások: Firestore (Google), Azure Cosmos DB
- Adattavak (Data Lake)
- Data Lakehouse
 - Ötvözik az adattavak és az adattárházak előnyeit, pl. Databricks, Snowflakes
- Adatstream feldolgozás
 - Pl. Kafka, Flink

- **AI által támogatott adatbázisok:** pl. **Oracle Autonomous Database**, amely mesterséges intelligenciával optimalizálja az adatbázis teljesítményét.
- **Generatív AI és SQL:** új AI eszközök (pl. ChatGPT) egyre jobban képesek SQL optimalizálásra és lekérdezések generálására.

Alapfogalmak - Adatbázis

Adatbázisnak nevezzük az egyedeknek és kapcsolataiknak valamilyen adatmodell szerinti elrendezését

Alapfogalmak – Adatbáziskezelő rendszer

Programok olyan gyűjteménye ami lehetővé teszi a felhasználók számára adatbázisok készítését és fenntartását.

Fontosabb feladatai:

- Adatok logikai és fizikai tárolása, karbantartása
- Adatok megjelenítése (lekérdezése)
- Egyidejű hozzáférések kezelése
- Jogosultságok kezelése
- Adatok mentése és helyreállítása
- Adatintegritás biztosítása
- Programozói és kommunikációs interfész biztosítása



ORACLE®

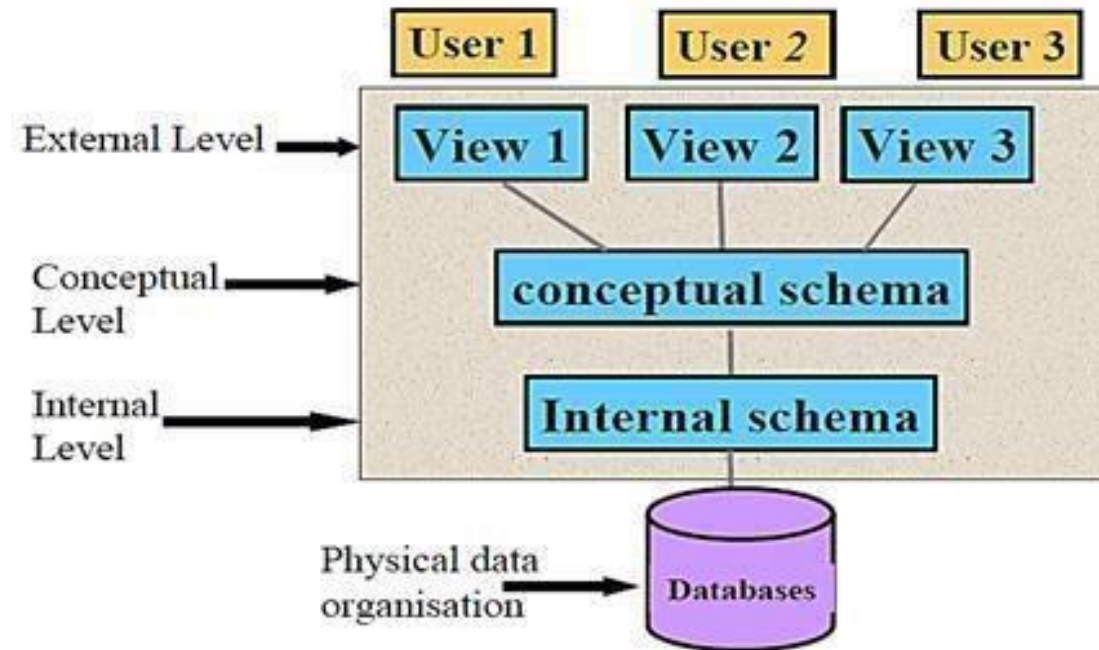


ANSI/X3/SPARC modell

Külső szint

Konceptcionális
szint

Belső szint



Az adatbáziskezelő
rendszer három szintre
osztja fel

Adatfüggetlenség:

Egy adott szinten történő változás nem érinti a felett lévő szinteket.

Pl: ha megváltoztatjuk a fizikai tárolás módját, az nem érinti a felhasználók lekérdezéseit

Alapfogalmak – Adatbázis rendszer

Az adatbáziskezelő rendszer és az adatbázisok együttese

DBMS + DATABASE



**Köszönöm
a figyelmet!**