Adatbázis kezelők programozási lehetőségei, standardok és fejlesztő környezetek

Miért kell programozni?

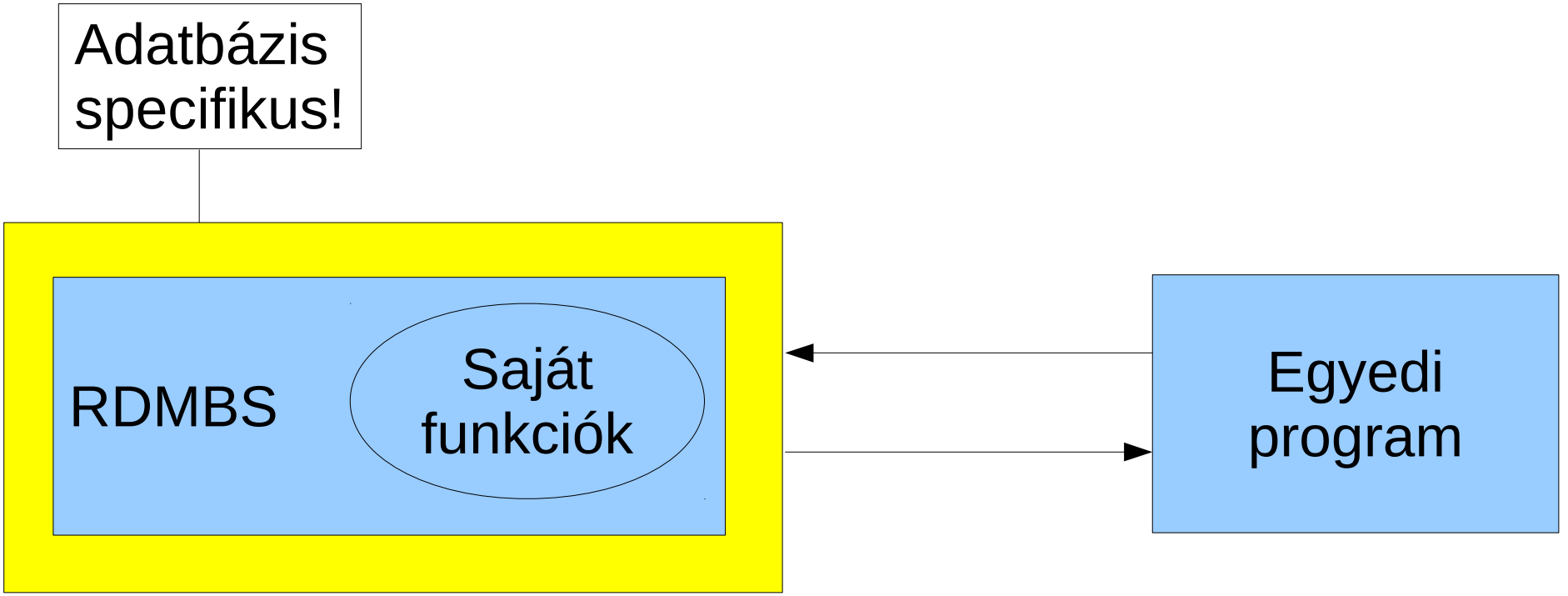
Nem várható el minden felhasználótól az SQL és az adatbázis séma ismerete, az SQL használata a profiknak is kényelmetlen lehet

Ismétlődő tevékenységek automatizálhatók
pl. havi jelentés készítés

Grafikus felhasználói interfész intuitívabb,
parancssor helyett menük, párbeszédablakok

Adatbázis-kezelő saját nyelvével

Az SQL mellett egy beépített procedurális nyelv
pl. PL/SQL, PL/pgSQL, VBA, ...
PL = Procedural Language (feltételek, ciklusok)
Az SQL nyelvet bővítik, a PL/xxx függvények az SQL utasításokból hívhatók

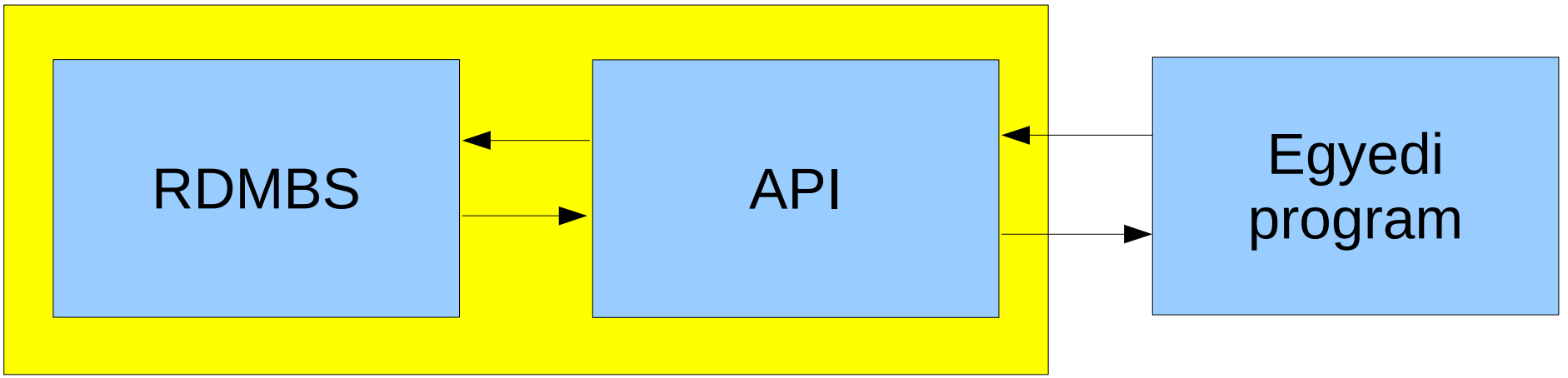


API – Application Programming Interface

Az egyes adatbázis-kezelőkhöz és programnyelv(ek)hez egyedileg kialakított program könyvtár

Az adatbázissal kapcsolatos tevékenységeket lefedi, a választott programnyelvből közvetlenül hívható eljárások

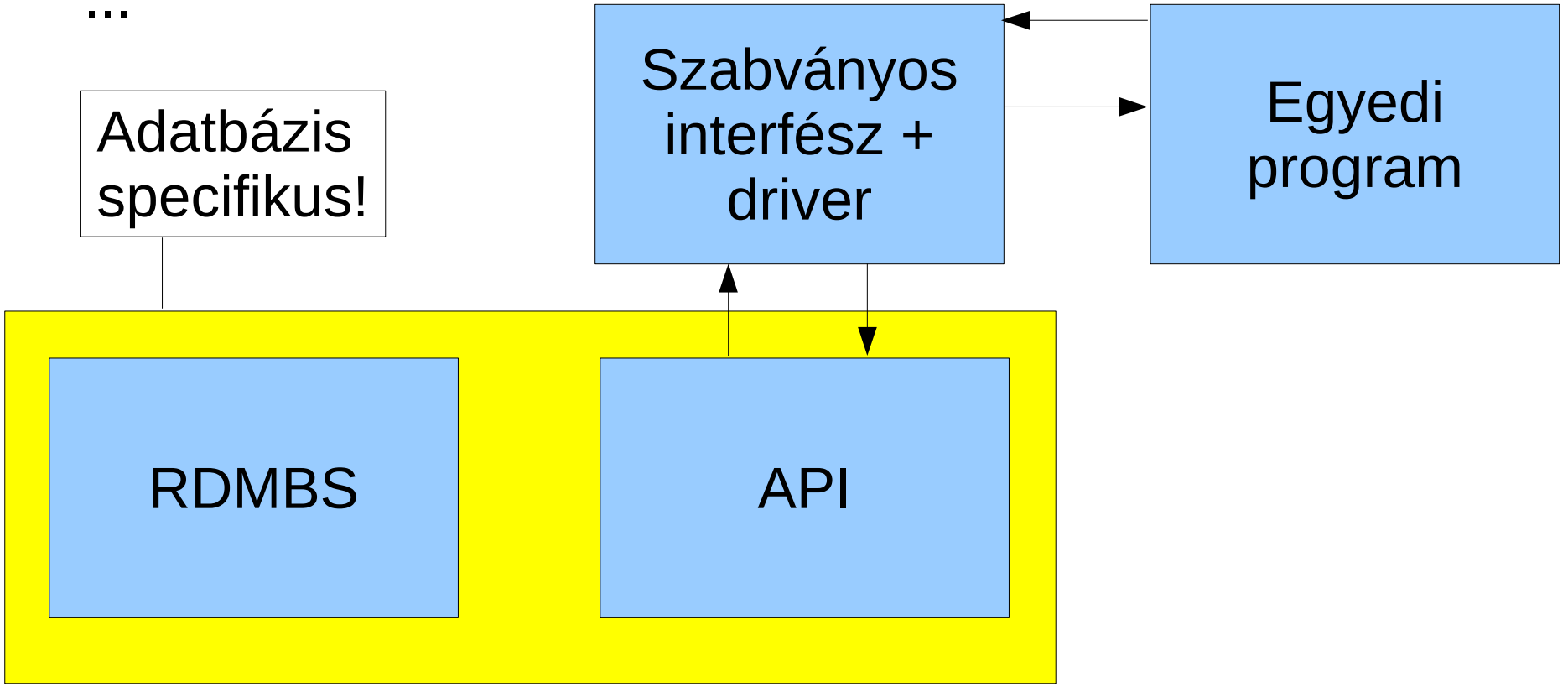
Általában az API függvények, objektumok nem szabványosak, nehéz váltani más adatbázis-kezelőre



Szabványos interfészek

Elfedik az eltéréseket az eltérő adatbázis API-k között

ODBC – Open Database Connection
JDBC – Java Database Connection
PDO – PHP Data Objects
Python DB API
ADO.NET – ActiveX Data Objects
...



Kapcsolódás az adatbázishoz

Connect string

adatbázis szerver (IP cím vagy DNS)
Port (szerver port pl. MySQL 3306)
felhasználó
jelszó
adatbázis

Példa PHP/MySQL:

```
$con = mysql_connect("localhost:3306",
"siki", "miki");
$result = mysql_select_db('minta', $con);
```

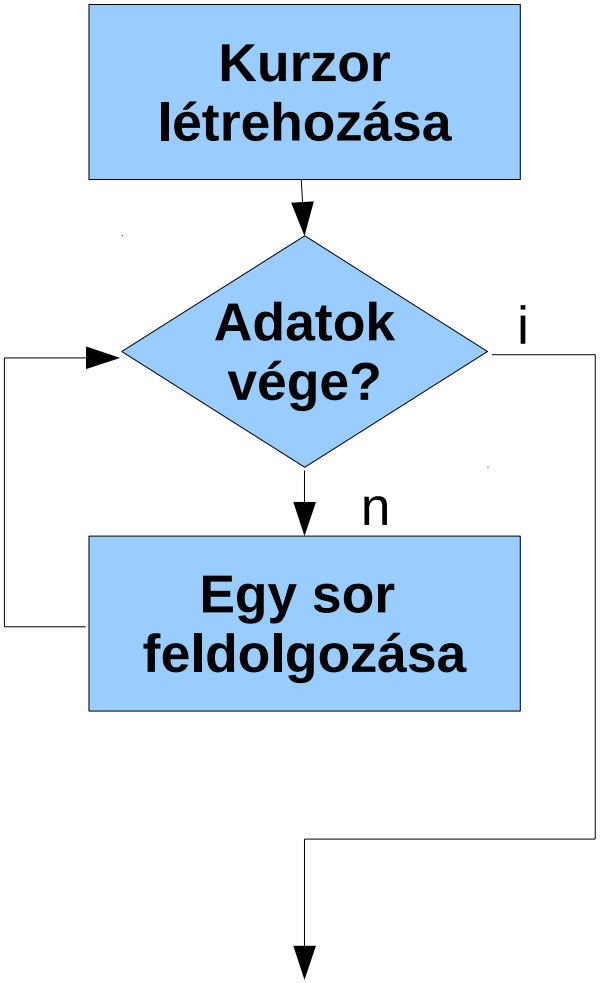
Példa C++/ODBC

```
HDBC lhdbc; HENV henv;
SQLAllocEnv (&henv);
retcode = SQLAllocConnect (henv, &lhdbc);
retcode = SQLDriverConnect (lhdbc, pWnd->m_hWnd,
        (UCHAR *) LPCSTR (dsn), (SWORD) dsn.GetLength (),
        (UCHAR *) con.GetBuffer (501), (SWORD) 500, &len,
        SQL_DRIVER_COMPLETE)
```

Kurzorok

Egyes lekérdező műveletek előre nem megjósolható számú adatot adnak vissza
A lekérdezés eredményét kisebb egységekben pl. soronként szeretnénk visszakapni
SQL:2003 CURSOR SQL szinten

```
DECLARE neves CURSOR FOR
  SELECT * FROM osztaly ORDER BY oszt_id
OPEN cnev
…
FETCH cnev INTO
…
CLOSE cnev
```




```
PHP/MySQL
$query_str = "select * from osztaly order by oszt_id";
$result = mysql_query($query_str);
while ($row = mysql_fetch_array($result)) {
      echo $row['oszt_id'], $row['nev'];
}
```

Adatbányászat

Adatok közötti előre nem ismert összefüggések feltárása

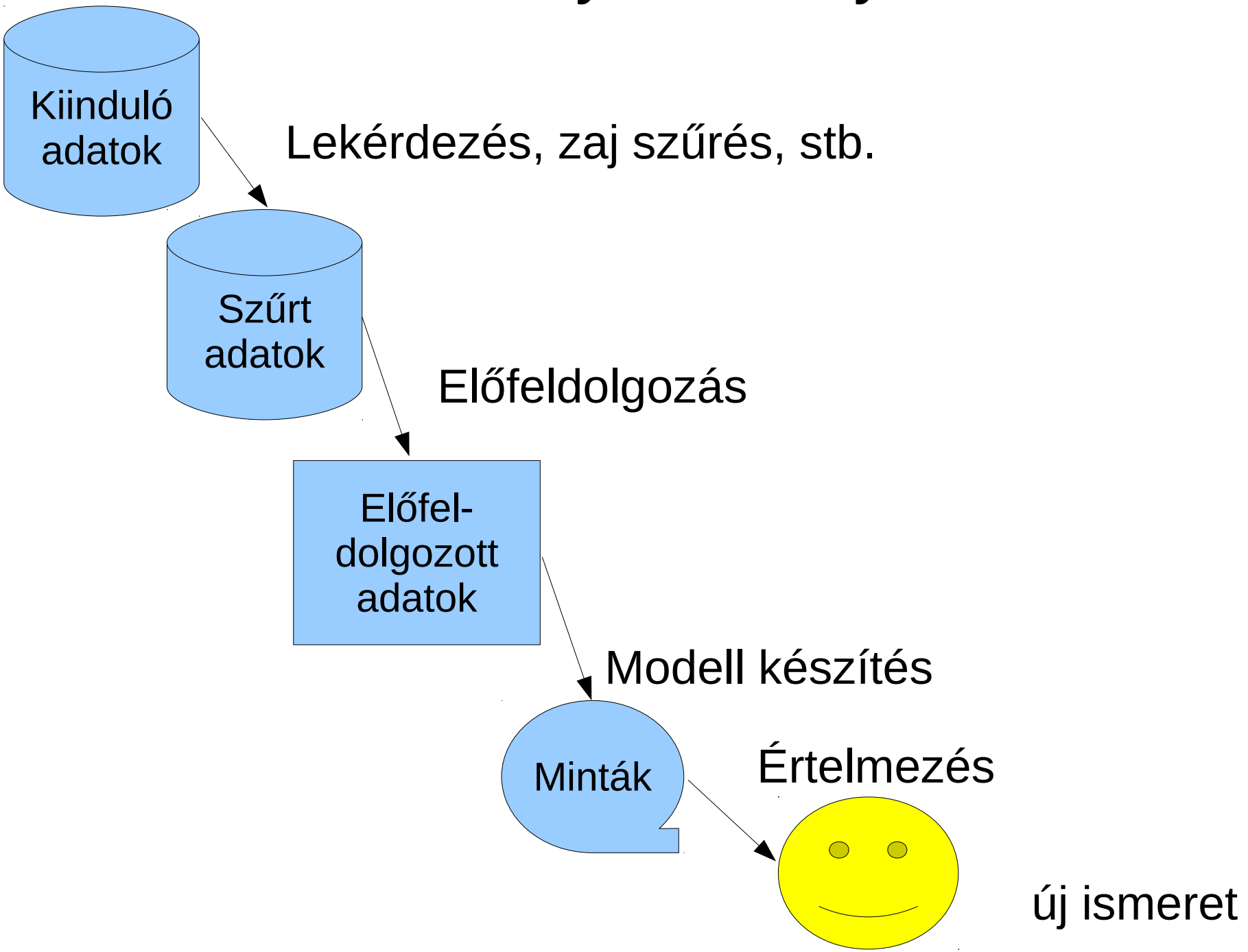
Négy általános feladattípus
Osztályozás – adatok besorolása előredefiniált csoportokba, pl. egy email spam-e? (döntési fák, neurális hálózatok)

Klaszterezés – csoportokba sorolás, de a csoportok előre nem adottak, pl. természetes törések

Regresszió – adatok modellezése függvénnyel (LKN)

Összerendelési szabály tanulás – változók közötti kapcsolat Keresés, pl. mely könyveket vásárolnak együtt a Könyvesboltban (tegyük egymás mellé őket)

Adatbányászat folyamata



Példa

Pl. van-e kimutatható kapcsolat a mechanika és matematika osztályzatok között?
Kiinduló adatok – Neptun adatbázis
Szűrés - legutolsó öt év adatai
Előfeldolgozás - akiknek nincs jegye mindkét tárgyból kimaradnak
Modell készítés – lineáris regresszió, korrelációs együttható
Értelmezés – mecha osztályzat = 0.7 * matek + 0.2

Következmény: elég az egyik tárgyból vizsgázni, ha szoros a kapcsolat (pl. r > 0.7) 

Térbeli adatokra - GIS