# *Adattárház, OLAP rendszerek*

## OLAP rendszerek fogalma

A korszerű információs rendszerek közös jellemzője, hogy nagy mennyiségű, különböző strukturáltságban tárolt adathalmazzal dolgoznak. Az adatok kezelése során azonban több, a rendszer működésére vonatkozó követelményt is figyelembe kell venni, melyek megvalósítására a rendszer implementálása során gondosan ügyelni kell. E követelmények közé tartoznak többek között az alábbi megkötések:

- A rendszernek mindig biztosítani kell az adatrendszer konzisztenciáját, végig teljesülniük kell. Egy ilyen szabály lehet például, hogy a csak attól a vevőtől fogadható el új rendelés, aki az előző rendeléseket mind kifizette.
- Gondoskodni kell arról, hogy az adatokat párhuzamosan többen is kívánják majd használni. Ebben az esetben meg kell oldani, hogy ne zavarják, rontsák el egymás adatait a párhuzamosan dolgozó alkalmazások.
- Biztosítani kell az adatrendszer védelmét is. Ehhez minden adatelem mellé fel kell jegyezni, hogy mely felhasználók mely műveletekre jogosultak az adott adatelemet illetően.
- Az adatrendszer adatvesztés elleni védelmi is fontos szempont. A rendszernek gondoskodnia kell arról, hogy egy esetleges rendszerösszeomlás, felhasználói tévedés esetén a lehetőséghez mérten minél kevesebb adat vesszen el.

A fenti kívánalmak mellett még számos olyan megkötésnek kell teljesülnie az információs rendszer adatkezelésénél, amelyek teljes megvalósítása komoly programfejlesztési kapacitást és erőráfordítást jelent. E követelmények maradék nélküli megvalósítására kidolgozott rendszerek az *adatbázis kezelő* (DBMS) rendszerek. Az adatbázis kezelő rendszerek alkalmazása esetén az adatok adatbázisba szervezetten kerülnek letárolásra. Az *adatbázisban* az információs rendszerhez tartozó minden adat perzisztens módon, azaz hosszú idejű tárolást megvalósító módon kerül elhelyezésre. Az adatbázisban az adatértékek mellett az adatok közötti kapcsolatok is megőrzésre kerülnek a kisegítő, adminisztratív célokat szolgáló adatokkal együtt.

Az adatbázis kezelő rendszer funkcionalitását, az elvégezhető műveletek körét, az adatok és kapcsolataik leírásának módját az adatbázis kezelő rendszer adatmodellje határozza meg. Az adatbázis kezelés fejlődése során néhány domináns adatmodell alakult ki, melyek között a legismertebb a relációs adatmodell, melyben az adatok több, egyenrangú táblázatban foglalnak helyet. A 1971-ben megalkotott *relációs adatmodellre* épülő adatbázis kezelő rendszerek (RDBMS) a leggyakrabban alkalmazott rendszer. Az RDBMS-re épülő klasszikus alkalmazások, melyek az 1980-as évek közepe óta egyre nagyobb számban terjedtek el, egyik fő jellemzője, hogy adminisztrációs célzatú rendszerek voltak, melyekben az operátorok segítségével vitték fel a valóságban bekövetkező eseményekhez, mint például egy helyjegy lefoglalásához tartozó adatmódosításokat. Az ilyen jellegű alkalmazásokat nevezik *OLTP*, azaz *on-line tranzakció orientált* alkalmazásoknak.

Az OLTP elnevezésben a tranzakció kifejezés adatbázisbeli értelmezéssel szerepel, mely szerint a *tranzakció* egy egységként kezelt több lépést is magába foglalható műveletsort jelent. A tranzakció kezelés egyik fő jellemzője az atomiság elve, ami arra utal, hogy a műveletsor

vagy teljesen végrehajtódik, vagy sehogy sem, mintha egyetlen egy eleme sem hajtódott volna végre.

Az OLTP rendszerek legfontosabb tulajdonságai a következőkben foglalható össze.

- Az adatkezelés adatmódosítás orientált, az alkalmazások zömmel az adatértékek megváltoztatásának szándékával kapcsolódnak az adatbázishoz. Az operátorok a valóságban bekövetkezett eseményeket rögzítik az adatbázisban, az alkalmazások célja lehetőséget adni a változások adatbázisba történő felvitelére.
- Az adatbázis a modellezett rendszer aktuális állapotát tartalmazza. Egy helyjegyfoglaló rendszer esetében például az éppen foglalt és szabad helyeket tartalmazza az információs rendszer adatbázisa. Az adatbázis tábláiban minden egyednek az éppen érvényes állapota, értéke tárolódik. Az adatok tárolásánál fontos, hogy az egyes egyedek egyértelműen meghatározhatók legyenek. A helyjegy foglalás esetében például fontos, hogy tudjuk mely helyre és esetleg ki által történt a foglalás.
- Nagy az egyes alkalmazások közötti konkurencia, ugyanis egyidejűleg több alkalmazás is kapcsolódik az adatbázishoz. Egy banki információs rendszer esetében például több ügyfél is vehet ki pénzt párhuzamosan a különböző banki automatáknál.
- Lényeges az adatrendszer konzisztenciájának megőrzése. Fontos, hogy egyik alkalmazás se tudja elrontani az adatok helyességét, épségét. Ha valamely alkalmazás megpróbálná elrontani az adatok épségét, az adatbázis kezelő rendszernek vissza kell utasítani ezen műveletre vonatkozó igényeket.
- Rendszerint homogén adatforrással dolgoznak az alkalmazások. Ez azt jelenti, hogy minden kezelt adat rendszerint egyetlen adatbázis kezelő mögött kerül letárolásra, minden adat azonos adatmodell szerint kezelhető és azonos csomóponton foglal helyet.
- Az adatok normalizáltan kerülnek letárolásra. Ez azt jelenti, hogy az adatbázis megtervezése során az egymástól függetlennek tekinthető adatelemek külön táblázatba kerülnek. Vagyis a logikailag, a felhasználó szemszögéből összetartozónak vélt adatelemek szétszórtan helyezkedhetnek el az adatbázisban. E szétdarabolásnak az adatok integritásának, az adatmódosítások kezelésénél van jelentősége, előnye.
- Ugyan a relációs adatmodell igen rugalmas és hatékony lekérdező nyelvet biztosít, ez a rugalmasság elsősorban a hagyományos programozói felületekhez viszonyított rugalmasságot jelent. Egy normál, programozói ismeretekkel nem rendelkező felhasználó számára a relációs adatmodell lekérdező felülete igencsak bonyolultnak tűnhet. Ezért megfelelő előképzettség nélkül nem lehet rugalmas információ lekérdezést elvégezni.
- Ezen alkalmazásokban igen fontos az adatvesztés elleni védelem megbízható megvalósítása. Egy banki rendszer esetében például igen fontos, hogy semmilyen esetben sem törlődjenek az aktuális adatok, hiszen azok máshonnan nem érhetők el és a jövőbeli adatok is ezen adatokra épülnek.
- A rendszert módosító alkalmazások zömében rövid futási idejűek. Egy banki tranzakció esetében például a tranzakció az azonosítást, a számla állás változtatás felvitelét, a naplózást foglalja magába. Ezen műveletek időszükséglete néhány percre tehető.

Az OLTP rendszerek igen elterjedtek mai is, hiszen az alkalmazások döntő többsége ilyen jellegű feladatokat lát el. Mára az OLTP rendszerek érett, kiforrott technológiát képviselnek, melyek az alkalmazások számára a következő előnyöket nyújtják:

- Hatékony adatkezelés. Az OLTP rendszerek igen gyors belső végrehajtó motorral rendelkeznek, mely lehetővé teszi a nagyobb adatmennyiségekben való művelet végrehajtást elfogadható válaszidőn belül.
- Az OLTP rendszerek rugalmas tervezési és kezelési felülettel rendelkeznek, melyek megkönnyítik a változások nyomon követését és realizálását.
- Biztosított a tranzakció kezeléstől megkövetelt integritási elvek betartatása.
- Létezik egy szabványkezelő felület és parancsnyelv, melyet megismerve a felhasználó vagy programozó több különböző rendszerrel is képes lesz munkát végezni.

Az alkalmazásoknak az előbb említett fő csoportja mellett van azonban egy olyan szelete is, amelynek jelentősége napjainkban egyre növekszik, amelyeknél a fenti tulajdonságok megléte már nem elegendő a hatékony működéshez. Itt gondolhatunk például egy olyan alkalmazásra, melyben egy vállalati vezetés számára kell különböző mélységű és különböző tartalmú jelentéseket készíteni a vállalat eddigi teljesítményére, tevékenységére vonatkozóan.

Ebben az esetben az alábbi nehézségekkel találná magát szembe a rendszer programozója, alkalmazója:

- A lekérdezések tartalma gyakran változhat, nem célszerű azokat előre, a rendszer fejlesztésének időpontjában lerögzíteni. Ezért a rendszernek olyannak kell lennie, hogy a felhasználó is meg tudjon fogalmazni a gép által érthető módon kérdéseket. Erre a feladatra az OLTP rendszerek nem alkalmasak, hiszen a lekérdező felületük a relációs algebrán alapszik, mely viszont megfelelő számítástechnikai előképzettséget igényel. Az igényelt lekérdezői felületnek alkalmasnak kell lennie a múltbeli adatokra is építő műveletekre, hiszen a jövőben várható folyamatok meghatározásánál a jelen állapot mellett a múltbeli állapotok is jelentős szerepet játszanak.
- Az OLTP rendszerekben a normalizált adattárolás következtében az egyes összekapcsolódó információ elemek szétdaraboltan, több különböző táblázatba szétosztva helyezkednek el. A megfelelő eredmény eléréséhez a felhasználónak ismernie kellene az adatmodell pontos struktúráját, s a létrehozott adatmodellt, a megalkotott táblázatok nevét és rekordszerkezetét. Ezen nagyobb információhalmaz ismeretét azonban nem szabad elvárni egyetlen felhasználótól sem.
- Az OLTP rendszerek rendszerint egy szűkebb problémakör adatait fedik le, s mint említettük, elsődlegesen az aktuális adatok tárolására szorítkoznak. Sok esetben viszont olyan információkra van szükség, melyeket több különböző, sok tekintetben egymástól független rendszerből kell összeszedni, illeszteni. Ehhez valószínűleg, több, esetleg eltérő adatmodellel, formátummal rendelkező adatforrást kell felkeresni az alkalmazásnak. A helyzetet bonyolíthatja, hogy ezen adatforrások adattartalma nincs szinkronizálva, hiszen korábban önállóan fejlődtek, működtek, ezért a rendszernek az adatok puszta átemelése mellett sok esetben még egyfajta illesztési munkát is el kell végeznie.

Mindezek a korlátok azt mutatják, hogy egy rugalmas, komplex lekérdezéseket támogató, emberközeli kezelő felülettel rendelkező információs rendszer megvalósításához a meglévő

OLTP lehetőségek nem a leghatékonyabb megoldást kínálják. Mivel ezen alkalmazási terület igényei, elvárásai lényegesen eltérnek a hagyományos OLTP alkalmazások jellemzőitől, ezért ezen alkalmazási területnek egy új elnevezést is adtak, mely tény már önmagában is mutatja, hogy itt egy lényeges új és fontos területről van szó, melyre igen komolyan oda kell figyelni. Ezen új alkalmazási terület az OLAP elnevezést kapta. Az *OLAP* rövidítés mögött az *on-line analitikai*, elemző folyamatok (on-line analitical processing) jelentés húzódik meg. Ezek az alkalmazásokhoz tehát információ feldolgozási, elemzési feladatok kapcsolódnak, melyeket rendszerint valamilyen döntéstámogatási cél érdekében kell elvégezni.

Az OLAP jellegű alkalmazások iránti igény azonban nem újkeletű, hiszen már a 70-es évek közepe táján készítettek olyan kísérleti alkalmazásokat, melyekben megpróbáltak egy rugalmas, felhasználóbarát kezelő felülettel rendelkező, a különböző szervezetek vezetői számára, a stratégiai döntések meghozatalát segítő rendszert megvalósítani. Mivel azonban a felmerült feladatok hatékony megvalósítása csak megfelelő hardver és szoftver támogatással biztosítható, ezért csak napjainkra vált lehetővé az elképzelések szélesebb körben való megvalósítása.

A konkrét tevékenységek szintjén nézve míg az OLTP rendszer segítségével fel tudjuk jegyezni a valóságban bekövetkező változásokat, mint például azt megvettek egy adott terméket egy adott boltban egy magadott napon, s így az OLTP rendszerem tárolni fogja minden egyes eladás adatát, vele együtt azt is, hogy például mennyi fogyott az egyes termékekből a különböző napokon és boltokban. Ezen információk hatékony és rugalmas lekérdezésére viszont az OLTP rendszerek helyett az OLAP rendszerek adnak jobb megoldást. Az OLAP rendszert használva például választ kaphatok az alábbi kérdésekre a letárolt vásárlási adatokra vonatkozólag:

- sikerült elérni a kitűzött célokat, mutatókat az eladások, a forgalom terén,
- sikeres volt-e a bevezetett kedvezményes akció,
- hogyan alakult az egyes termékkategóriák forgalma az elmúlt hónapokban,
- milyen forgalmi adatok várhatók a következő időszakban,
- mely termékeket lehet összekapcsolni az akcióknál,
- mely boltoknál volt a legnagyobb eltérés az általánosan lezajló folyamatokhoz képest a forgalom tekintetében.

E kérdések körét természetesen még sokáig lehet bővíteni a felsoroltakhoz hasonló újabb kérdésekkel.

Az OLAP rendszerek legfontosabb tulajdonságait a következőkben foglalhatjuk össze.

- Az adatkezelés adatlekérdezés orientált, az alkalmazások zömmel az adatértékek lekérdezésének szándékával kapcsolódnak az adatbázishoz. A végrehajtandó lekérdezések egyik fő jellemzője, hogy azok elemzési jellegűek, így sokkal nagyobb adatmennyiséget is érintenek, mint az OLTP rendszerekben szokásos lekérdezési műveletek. Rendszerint, a nagyobb adatmennyiséget megmozgató, komplexebb számításokat magukba foglaló utasítások időszükséglete is sokkal jelentősebb mint az OLTP rendszert esetén. Az OLAP rendszerekben nem ritkák a több órás válaszidővel rendelkező lekérdezések sem.
- Az adatbázis a modellezett rendszer aktuális állapota mellett a múltbeli adatokat is tartalmazza. Egy vállalati rendelési információs rendszer esetében például az éppen élő rendelések mellett a korábbi időszakok rendeléseit is tartalmazza az információs rendszer adatbázisa. Az adatbázis tábláiban minden egyednek az éppen érvényes állapota, értéke mellet azok története is tárolódik. A múltbeli

adatok segítségével, felhasználásával pontosabb lehet az elemzési munka, a jövő előrejelzése.

- Kicsi az egyes alkalmazások közötti konkurencia, ugyanis egyidejűleg csak egy vagy néhány alkalmazás kapcsolódik az adatbázishoz. Ennek oka, hogy az OLAP rendszerek zömében a vezetői, menedzsment réteg számára készül, ahol viszonylag szűkebb a rendszerhez hozzáférési jogokkal rendelkező felhasználók száma, másrészt a lekérdezések nem a napi munka irányítására vonatkoznak, s csak ritkábban van szükség a végrehajtásukra.
- Mivel az adatrendszer elsődleges lekérdezési funkciót szolgál ki, s ritka a módosítási művelet, ezért itt nem az adatrendszer konzisztenciájának megőrzése nem jelenti a legfontosabb feladatot. Ezzel szemben viszont fontos, hogy a rendszer a fellépő lekérdezési műveleteket a lehetőségekhez mérten minél rövidebb idő alatt végrehajtsa. Ehhez az OLTP rendszertől eltérő optimalizálási modulok, módszerek beépítése van szükség az OLAP rendszert megvalósító rendszerekben.
- Az OLAP alkalmazások egyik fontos jellemzője, hogy több különböző, inhomogén adatforrással dolgoznak. Ez azt jelenti, hogy a műveletbe bevont adatok több különböző adatbázis kezelőből kerül beolvasásra, az egyes adatelemek különböző adatmodell szerint tárolva és különböző csomóponton foglalhatnak helyet.
- Az adatok belső tárolási formátuma még jobban rejtve marad a felhasználók előtt. Fontos, hogy a felhasználói lekérdezői felületnél az adatok megtartják mindazon kapcsolataikat, strukturáltságukat, amelyek a felhasználókban természetes módon megjelennek. Így a felhasználó a kérdéseit a hozzá közel álló formalizmusban és jelentéssel teheti fel, s nem kell elsajátítania egy gépközeli formalizmust. Az OLAP rendszereknek tehát egy hatékony, rugalmas, felhasználói szemlélethez közel álló lekérdezői felülettel kell rendelkezniük.
- Az OLAP alkalmazások az adataikat más, rendszerint OLTP rendszerekből veszik át. Az OLAP rendszer tehát nem maga az adatforrás, ez inkább egy adatintegráló modul. Mivel azonban az OLTP rendszerek rendszerint csak az aktuális adatokat tárolják, ezért az OLAP rendszernek kell gondoskodni arról, hogy a múltbeli adatok is rendelkezésre álljanak a feldolgozási műveleteknél. Ehhez az OLAP rendszernek az OLTP-től átvett adatokat meg kell őriznie, így az OLAP is adattárolási funkciót megvalósító rendszerré válik, s ezáltal itt is fontos az adatvesztés elleni védelem megvalósítása. Természetesen, a jóval kisebb gyakoriságú módosítási arány miatt ez most egyszerűbb mechanizmusokkal is megvalósítható.
- A rendszerben kapcsolódó alkalmazások zömében hosszabb futási idejűek, mivel az igényelt lekérdezési, elemzési műveleteket rendszerint igen összetettek, komplexek. Egy beruházási döntést előkészítő elemzési feladatnál például szükség lehet az előző nyolc év forgalmi adatira a különböző régiókra és időszakokra vetítve, mely során a jövőre vetített trendek meghatározása is megvalósulhat.
- Hatékony lekérdezés megvalósítások. Ugyan összetettebb lekérdezések jellemzik az OLAP rendszereket, a felhasználók szemszögéből nézve viszont ugyanolyan hatékonysági elvárások élnek mint bármely információs rendszerrel szemben, ezért az OLAP rendszerek megvalósítása esetén különös gondot kell fordítani a hatékony válasz generálási algoritmusokra a nagyobb adatrendszerek esetében is. Ezen cél elérésére például a rendszer előre letárolja a különböző előszámítások, részműveletek eredményeit.

- Az OLAP alkalmazások az egyszerűbb lekérdezések helyett összetett elemzési funkciókat megvalósító lekérdezéseket tartalmaznak. Ezen elemzési funkciók a nagy adathalmazból összesítő, aggregált, szabályok és trendek feltárására alkalmas adatokat állítanak elő.
- Az OLAP rendszereket is egyidejűleg többen használhatják, igaz alapvetően mindegyik hozzáférés olvasás jellegű. Ezáltal az OLAP is egy osztott adatforrással dolgozik. A megosztás megtervezésekor azonban azt sem szabad figyelmen kívül hagyni, hogy hatékonysági megfontolásokból kiindulva egy OLAP adatforráshoz a hasonló jellegű alkalmazásokat célszerű hozzákapcsolni.
- Az OLAP által kezelt adatok legnagyobb mértékben felhasználó barát megjelenítési és tárolási formája a multidimenzionális adatmodellre épül. A multidimenzionális adatmodell fő jellemzője és ereje, hogy az adatokat több más adatelemtől, a dimenzióktól való függőség szerint ábrázolja. S egy mennyiséget tetszőleges sok más dimenzió függvényében tudja ábrázolni.

Az OLAP rendszerek legfontosabb jellemzőinek összefoglalására tett egyik legkorábbi és legismertebb javaslat a Codd által definiált, s 1993-ban megjelent tulajdonságlista. E kritériumok célja annak specifikálása, hogy mikor tekinthető egy rendszer OLAP rendszernek. Codd kritériumai iránymutatóként alkalmazhatók az OLAP rendszerek azonosításában, e kritériumok jelentőségét azonban csökkenti az a tény, hogy igen sok közöttük a termék specifikus tulajdonság, melyek a gyakorlatban nem minden OLAP rendszerben került a megadott módon megvalósításra. Codd szabályai:

- az adatrendszer a multidimenzionális adatmodellen alapszik
- felhasználóbarát adat kezelő felület megléte
- az OLAP rendszer bemenete heterogén forrásadatok rendszere, s kimenete felhasználóbarát elemzési modulok
- rugalmas adatbetöltési funkciók biztosítása egy köztes tárolási szinten keresztül
- széles körű adatelemzési funkciók biztosítása köztük változatok képzése s elemzése
- a rendszer a kliens-szerver struktúrán alapuljon
- transzparens hozzáférés biztosítása az OLAP adatokhoz
- több felhasználó konkurens hozzáférése biztosított
- nem normalizált adatok kezelése
- OLAP eredmény adatok éles elkülönülése a forrás adatoktól
- A hiányzó adatok jelölésére szolgáló NULL érték nem normál adatérték
- A hiányzó adatokat az elemző rendszerek nem veszik figyelembe
- Rugalmas jelentés készítési lehetőségek állnak rendelkezésre
- A jelentés készítés hatékonyságát ne befolyásolja a felhasznált dimenziók darabszáma
- Rugalmas és optimalizált fizikai tárolási struktúra, mely képes illeszkedni, alkalmazkodni a változó körülményekhez
- Minden dimenzió egyenrangúan kezelendő
- Tetszőleges dimenziószám és aggregációs szint biztosítása

Ugyan az OLAP rendszereket a hagyományos, OLTP igényekhez szabott adatbázis kezelő rendszerekre alapozva is meg lehet valósítani, az így létrejövő rendszerek hatékonysága, rugalmassága nem tekinthető optimálisnak. Ennek fő oka, hogy a hagyományos adatbázis kezelő rendszerek belső motorja az OLTP igényekhez hangolt, az ott lezajló műveletre optimalizált. Egy hatékony OLAP rendszerhez egy megfelelő, az OLAP igényekhez igazított adatbázis kezelő rendszerre lenne szükség. Mivel ez az igény már viszonylag régóta

megjelent, s a technikai lehetőségek is rendelkezésre állnak, minden adott volt ahhoz, hogy ténylegesen is létrejöjjenek az OLAP igényekhez igazított adatbázis kezelő rendszerek. Ezen speciális adatbázis kezelő rendszereknek, a hagyományos adatbázis kezelő rendszerektől való megkülönböztetésként új névvel keresztelték el. Ezen adatkezelő rendszereket hívják *adattárházaknak* (*DW*, data warehouse).

## Adattárházak fogalma és struktúrája

Az adattárházak ötlete, mint már említettük, nem mai eredetű, egészen a 80-as évek végéig nyúlik vissza, amikor Inmon nevéhez kapcsolódóan elindult az adattárházak kialakulása. Az adattárház alapdefiníciója is Inmon-tól ered, mely szerint az adattárház "téma orientált, integrált, az adatokat történetiségében tároló adatrendszer, amelynek fő célja az adatokból történő hatékony információ kinyerés biztosítása, elsősorban a döntéshozatali folyamatok támogatása céljából".

Az adattárházak jellemzésekor több különböző szemszögből is végezhetünk vizsgálatot. Ha a DW rendszerek főbb funkcióit vesszük sorra, akkor a következő elemekre érdemes kitérni:

- A DW több különböző, heterogén szerkezetű OLTP adatforrásból integrálhat be adatokat
- Az adattárházaknak gondoskodni kell a heterogén forrásokból bekerülő adatok formai egységesítéséről és a tartalmi integritás, ellentmondás mentesség biztosításáról.
- A DW rendszerekben a hagyományos OLTP rendszerekhez képest nagyságrenddel nagyobb adathalmaz tárolás és kezelést kell hatékony módon megvalósítani.
- Az adattárházba bekerült adatok elsődlegesen döntéstámogatási céllal kerülnek felhasználásra, vagyis a DW rendszerek a lekérdezési műveletek hatékony végrehajtására optimalizáltak
- Az adattárházak az adatokat egy felhasználóbarát, a felhasználó szemléletmódjához közelálló struktúrában tárolják.
- A DW rendszerekhez rugalmasan kapcsolódó lekérdező felületeknek támogatniuk kell a bonyolultabb statisztikai jellegű, elemzési és adatbányászási műveleteket is.
- Lehetőséget kell biztosítani az adattárház rendszerekben arra is, hogy a rugalmasan be lehessen állítani a működési környezet paramétereit is, melyek közé tartozik többek között például az adatbetöltés ütemezése, az adatok hozzáférési védelmének beállítása.
- A DW rendszerekben gondoskodni az adatok múltbeli verzióinak a megőrzéséről is, s az adatkezelő műveleteknél kiemelt fontossággal kell kezelni az időbeliségre vonatkozó elemek biztosítását is.

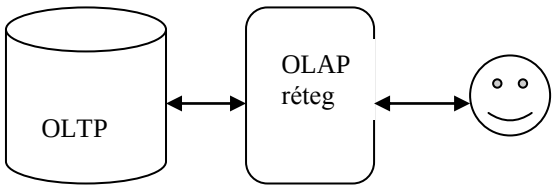
Az adattárházak vizsgálatakor egy másik lehetséges szempont a DW rendszer fizikai felépítésének, topológiájának a vizsgálata. Ennek során a DW rendszerbe bevont elemek elhelyezkedését s a közöttük fennálló kapcsolatokat kell vizsgálni. Első megközelítésben a DW rendszerhez kapcsolódóan három fő szereplőt célszerű megkülönböztetni

- OLTP adatforrások
- Maga a DW rendszer
- Felhasználói, kliens alkalmazások, segédeszközök

A kliens oldali segédeszközök szolgálnak arra, hogy a felhasználó az adattárház tartalmára vonatkozóan feltehesse kérdéseit, elemzési vagy adatbányászási műveleteket hajthasson végre, s azok eredményeit könnyen értelmezhető formában megkaphassa. E három komponenst véve alapul a következő fizikai topológia alaptípusokat szokás megkülönböztetni:

- virtuális
- centralizált
- kétszintű
- elosztott
- hibrid.

A *virtuális DW* rendszerben a virtuális jelző arra utal, hogy itt nincs is igazi DW rendszer a struktúrában. A DW által biztosított szolgáltatásokat ekkor vagy az OLTP rendszerek fölé, az azokhoz kapcsolódó réteg, vagy a kliens oldalon elhelyezett DW rendszert szimuláló réteg végzi el.

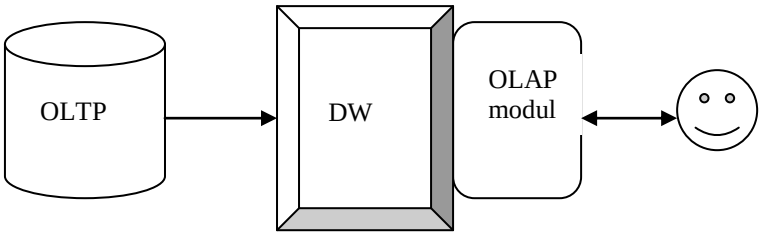



1. ábra, virtuális DW struktúra

Ezen megoldás előnye, hogy költségkímélő, hiszen egy sokkal egyszerűbb funkcionalitást biztosító OLAP elemet kell csak telepíteni, miközben az adatok továbbra is csak az OLTP adatbázisokban lesznek letárolva. Az olcsó megoldásnak természetesen számos korlátja van, mint például az, hogy

- korlátozott, az OLTP rendszertől függő az elérhető adatok köre,
- gyenge a rendszer optimalizálási képessége,
- korlátozott adatműveleti lehetőségek.

A *centralizált* elrendezés esetén már egy igazi központi DW működik, melyek köré kapcsolódnak mind a kliensek, mind az adatforrások.



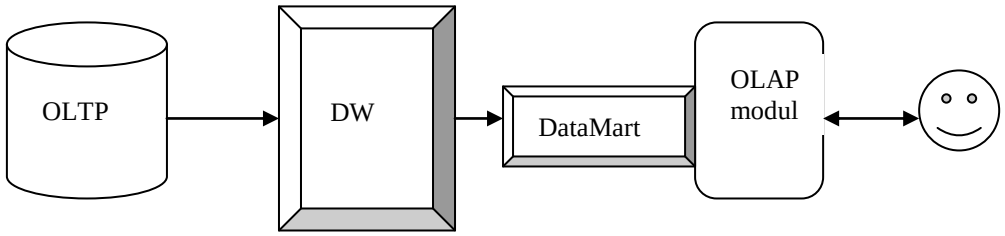

2. ábra, centralizált DW struktúra

Ezen elrendezés előnye, hogy egyszerűbben megvalósítható és karbantartható. Ekkor minden adat egy központi helyen helyezkedik el, így egyszerűbb lesz az adatok integrálásának feladata is. Ezen struktúrában minden egyes kliens a teljes adathalmazt elérheti közvetlen

módon is. Az egyszerűség viszont számos olyan problémát is magával hoz, ai a rendszer hosszabb távú működésének tervezése során mindenképpen figyelembe kell venni:

- A rendszer esetleges későbbi bővítésekor a centralizált séma fenntartása egy nagyobb beruházási költséggel valósítható csak meg, hiszen ilyenkor egy erősebb DW konfigurációt kell beszerezni a meglévő DW rendszer helyett.
- A rendszer kevésbé védett az esetleges működési zavarok ellen, hiszen minden DW funkció egy helyen valósul meg, s ha ez a csomópont kiesik, meghibásodik, akkor a teljes rendszer működése leáll.
- A rendszer teljesítményének javításakor viszonylag kevesebb lehetőség van a különböző optimalizálási lehetőségekre, hiszen csak egy DW elem végez adatkezelési feladatokat.

A fent említett hiányosságok kiküszöbölésére, a rendszer rugalmasságának fokozására irányuló leggyakoribb lépés a központi DW rendszer helyettesítése több DW funkciót ellátó DW egységgel. Ezen változtatások egyik lehetséges módozata a *kétszintű struktúra* lesz, melyben a központi DW rendszer mellé több, kisebb méretű és teljesítményű DW csomópontot telepítenek. Ezen kisebb DW testvéreket nevezik DataMart egységeknek




3. ábra, kétszintű DW struktúra

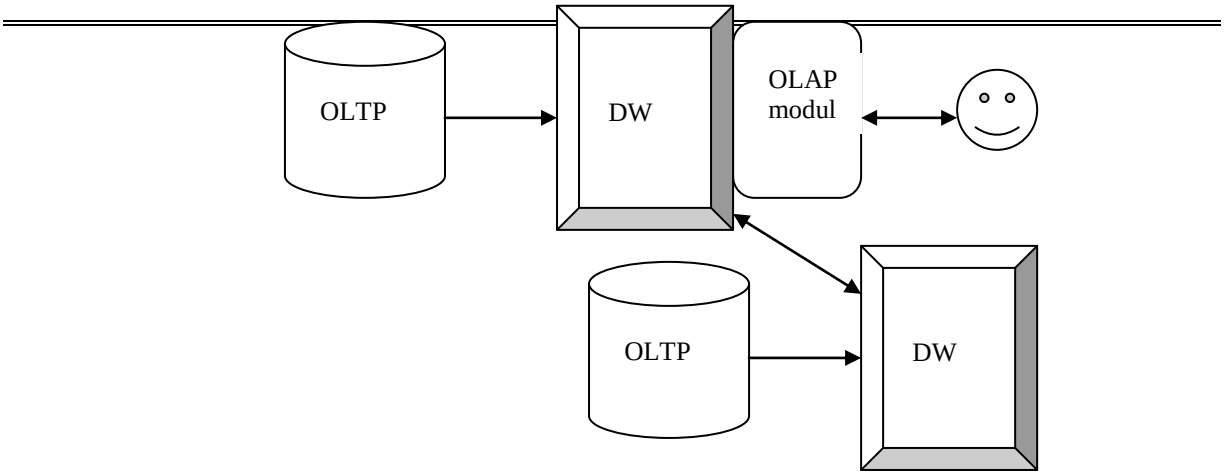
A DataMart rendszer alapvetően ugyanazon funkciók ellátására szolgál mint a DW rendszer, csak az átfogott, tartalmazott adatok mennyiségében, az átfogott problémakör nagyságában van különbség a két rendszer között. A DataMart rendszerint a vállalaton belül csak egy részleg, egy működési területet adatait ölel fel. A DataMart-hoz kapcsolódó alkalmazások csak az adott részterülethez kapcsolódó információkat érhetik el.

A DataMart-ok alkalmazásával lehetőség nyílik arra, hogy az adatkezeléshez kapcsolódó műveleteket most már ne csak egy hanem több egység is elvégezhesse, ezen munkamegosztás által növekedhet a rendszer teljesítőképessége is. Hasonlóan javítható a rendszer rugalmassága is, hiszen egy-egy DataMart egység beépítésével, amely költsége jóval kisebb egy erős DW egység telepítésének költségénél, igény szerint tovább növelhető a rendszer teljesítménye. Ebben a struktúrában az egyes kliensek csak a kapcsolódó DataMart-ok adatait érhetik el, habár bizonyos esetben a központi DW egységhez való kapcsolódás is megengedett. Ez a fajta elérhetőségi korlátozás az adatok védelmi rendszerének is biztos alapját képezheti.

A DataMart alapú struktúrában az OLTP adatok a központi DW rendszeren keresztül kerülnek át a DataMart egységekhez. Így egyetlen DW játsza továbbra is az adatgyűjtő szerepét. Ez az egyszerűség szempontjából előnyös megoldás, azonban a rendszer hatékonyságát károsan befolyásolja, ha egy olyan OLTP információs rendszert veszünk alapul, melyben az OLTP források egymástól igen távol helyezkedhetnek el. Ekkor az egyes OLTP adatok vándoroltatás a távoli DW csomópontokhoz jelentős hálózati költséget

jelenthet, hiszen itt valóban nagy mennyiségű adatok mozgatásáról van szó. Másrészt ebben az esetben továbbra is gyenge a rendszer hibatűrő képessége az adatok beolvasását illetően, hiszen egyetlen DW egység végezheti el csak ezen műveleteket.

A fenti problémák kiküszöbölésére alkalmas struktúra az *osztott elrendezés*. Ebben a struktúrában egy DW egység helyett több, a hálózat különböző csomópontjain elhelyezkedő DW rendszer foglal helyet.



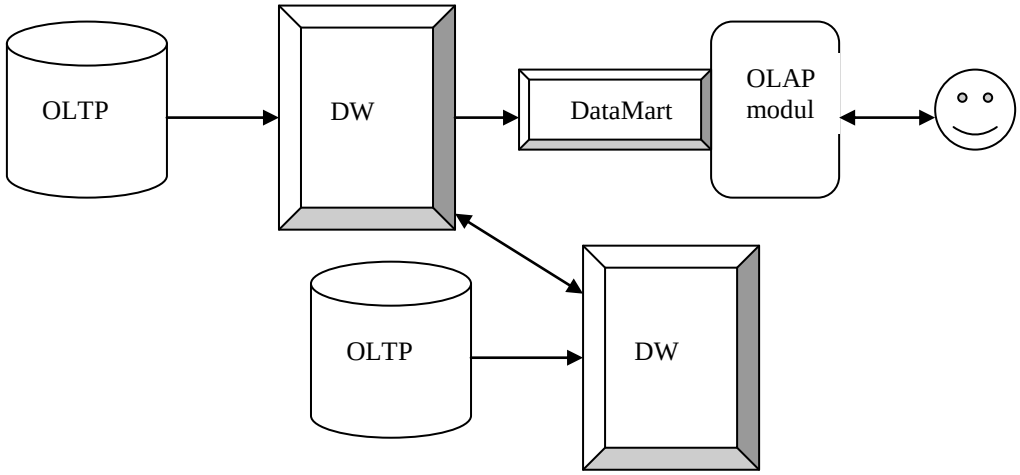

4. ábra, elosztott DW struktúra

Az elosztott topológia esetében a különböző OLTP források más-más DW csomóponthoz kapcsolódhatnak, s egyes behozott és átkonvertált adatok is más-más DW rendszerben kerülhetnek majd letárolásra. A betöltéshez szükséges hálózati forgalmat tehát sikerült így lecsökkenteni, azonban mint észrevehető, ezen topológia esetében viszont a lekérdezések megvalósításával lehetnek majd gondok. Ennek oka, hogy a lekérdezések az adattárház nagyobb adatszeletére is vonatkozhatnak, ha ez az adathozzáférés szempontjából megengedett, s ekkor előfordulhat, hogy a válasz generáláshoz szükséges adatokat most már több DW egységből kell összeszedni. Ez viszont azt jelenti, hogy most a lekérdezések során fog jelentősen megnőni a hálózati forgalom, mégpedig úgy, hogy ugyanaz az adatelemet kell majd átvinni többször egymásután is két DW csomópont között. Ezen felesleges adatforgalom redukálása céljából szokás olyan megoldást választani, melyben bizony az egyes DW egységek átadják adataik egy részét a másik DW egységnek az ottani letárolás céljából. Ezen adatok vonatkozhatnak a tényeleges, műveleti információkra és vezérlő információk is. Az elosztott rendszerekben tehát szokásos megoldás a hatékonyság és a hibatűrő képesség fokozása érdekében az adatok több helyen történő megismétlése, azaz replikációja.

A negyedik topológiai változat azt az esetet fedi le, amikor több DW és több DataMart egység is helyet kap a rendszerben. Ezen hibrid megoldást szemlélteti az 5. ábra.

Ezt a *hibrid* megoldás akkor szokott kialakulni, ha egyrészt több DW egység szükséges a sok távoli, szétszórtan elhelyezkedő adatforrás vagy a nagyobb rugalmassági követelmények miatt, másrészt ha az egyes kliensek hatékony és gyors működését több DataMart megvalósításával célszerű elősegíteni. Ekkor egy DataMart egység több helyről is olvashat be adatokat, azaz egy DataMart elérhet egy vagy több DW egységet, s emellett elérhet egy vagy több OLTP adatforrás egységet is. Ezzel a struktúrával elérhető egy optimális terhelés

kiosztás és egy rugalmas bővíthetőség. Másrészről viszont a heterogénabb környezet a rendszergazdák vállára fog nagyobb terhet és feladatot helyezni, hiszen meg kell oldania a heterogén komponensek illesztését, s több különböző részrendszer működését kell megérteni és vezérelni.




5. ábra, hibrid DW struktúra

A adattárházak működésének elemzésénél a fizikai elrendezés mellett egy másik fontos szempont a DW rendszer logikai struktúrájának a megismerése. A DW logikai modellje magába foglalja a DW rendszert alkotó komponenseket, s azok kapcsolatait. A következő ábrán a DW logikai modelljének legfontosabb elemei láthatók.


6. ábra, teljes DW rendszer logikai modellje

A modellben az alábbi funkcionális elemek foglalnak helyet:

- *Adatforrás réteg*: ez a réteg fedi le az egyes OLTP adatforrásokat, nem feledve, hogy az egyes adatforrások eltérő struktúrájúak, inhomogének lehetnek.
- *Adattovábbító réteg*: ebbe a rétegbe kerülnek be elsőként az adatforrásokból áthozott adatok. Itt történik az adatok egységes formátumra történő átalakítása, s az adatok tartalmi integritásának vizsgálata is, amit adattisztítási folyamatoknak is szokás nevezni. Ezen előkészítő területet nevezik Data Staging Area-nak
- *Adattárolási réteg*: ezen réteg gondoskodik a behozott, tisztított adatok lehelyezéséről, tárolásáról. Az itt megvalósítandó feladatok közé tartozik többek között a bejött adatelem beillesztése a meglévő struktúrába, vagy a meglévő optimalizálási elemek aktualizálása.

- *Adatszótár réteg*: az adatszótár a rendszerhez tartozó metaadatok tárolására szolgál. A metaadatok a normál, felhasználói adatokra vonatkozó adatokat foglalják magukba. Ide tartoznak többek között a struktúrát vagy a védelmet leíró információk is.
- *Ütemező réteg*: az adattárház belső karbantartási, adat betöltési folyamatainak automatizálására az DW rendszer rendelkezik egy aktív adatbázis funkciókat megvalósító modullal is. Az ütemező a beállított paraméterek alapján a háttérben dolgozva hajtja végre a feladatokat.
- *Adathozzáférési modul*: Az adattárházban tárolt adatok hatékony és egyszerű elérésére a DW rendszer rendelkezik egy adathozzáférési modullal is. Ebben a rétegben foglal helye a felhasználói parancsnyelv implementálása is. Az alkalmazások, kliensek ezen modulon keresztül férhetnek hozzá az adattárházban tárolt adatokhoz.
- *Információ megjelenítési réteg*: ez a réteg azon segédprogramokat öleli fel, melyek a felhasználó részére készültek és céljuk az adattárházban tárolt adatok könnyen értelmezhető, grafikus vagy táblázatos formában való megjelenítése. Ezen modul lehetővé teszi, hogy az alkalmazónak ne kelljen ismernie a DW rendszer konkrét parancsnyelvét, hanem a rendelkezésre álló segédeszközöket használhatja, melyek kezelése nem igényel számítástechnikai előképzettséget. Az információ megjelenítési réteg leggyakoribb elemei a következő segédprogramok:
  - Ad-hoc Query Tools: rugalmas lekérdezési lehetőségeket nyújtó program
  - Report Writers: nyomtatott, listás jelentések készítése
  - Forecasting Tools: előrejelzési elemzések
  - DSS: döntéstámogatási eszközök
  - Scoring Tools: helyzet értékelési eszközök
  - Data Mining: adatbányászási eszközök

Az adattárházak fenti szerkezeti váza egy olyan keretként értelmezendő, melyre a legtöbb DW rendszer struktúrája ráillik. Termeszétesen az egyes konkrét DW rendszerekben más és más lehet az egyes rétegek kiépítettsége, funkcionalitása és belső megvalósítása.

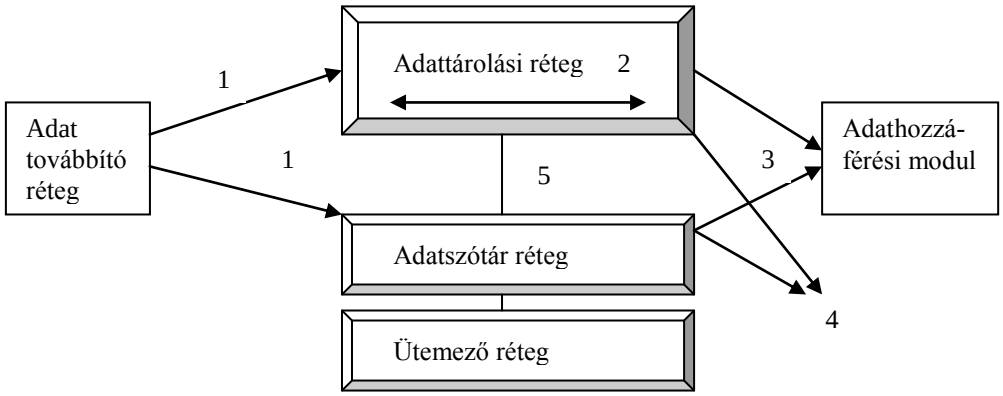
A strukturális, funkcionális oldal mellett célszerű az adattárház rendszereket egy másik fontos szempont, az adattárházakban megvalósuló adatfolyam szerint is rendszerezni. Az adatfolyam alatt az adatok mozgását értjük a DW rendszer különböző komponensei között. Az adatok mozgása a DW jellegéből következően többféle céllal is történhet, ugyanis a DW előbb be kell hozni az adatokat a külső adatforrásokból, majd fel kell dolgozni őket, mely során az adatok az adattárházból az alkalmazásokhoz kerülnek át. Az adatmozgás jellege, indítéka alapján az alábbi adatfolyam típusokat különböztetjük meg:

- *bementi adatfolyam* (1, inflow): ez az adatoknak a külső OLTP adatforrásokból történő átemeléséhez kapcsolódó adatmozgatást jelent
- *belső adatfolyam* (2, upflow): ez az adattárházon belül lejátszódó adatmozgatásokat jelenti, amikor is a beérkező adatokat a rendszer feldolgozza, átalakítja, hogy a későbbiekben hatékonyabban ki tudja majd szolgálni a beérkező lekérdezési műveleteket
- *kimenő adatfolyam* (3, outflow): ebben a fázisban az adattárházból az alkalmazások, a kliensek felé halad az adat. Ez az adathalmaz a klienstől érkező lekérdezési parancsra küldött választ foglalja magába.
- *selejtezési adatfolyam* (4, downflow): mivel az adattárházba bevitt adatok bizonyos része az idő múlásával elvesztik fontosságukat, ezért ezen adatelemek

nem célszerű továbbra is benn tartani az adatbázisban, vagyis szükség lesz a bevitt adatok kiemelésére is. Így a kiemeléshez, leselejtezéshez is kapcsolódik egy külön adatfolyam.

- *vezérlő adatfolyam* (5, metaflow): az adattárházban tárolt adatelemek leíró, kísérő információi is mozognak az egyes DW komponensek között, hiszen például egy elemzési segédprogramnak is tudnia kell az adatszerkezetre vonatkozó információkat, hogy könnyebbé tegye a felhasználó dolgát az adatrendszerben való navigáláskor. Ezen leíró adatok, metaadatok mozgása alkotja a vezérlő adatfolyamot.

A fenti adatfolyamokat szemléletes mutatja be a következő ábra, melyben az egyes adatfolyamok tipikus forrás és célhelye is fel van tüntetve.




7. ábra, DW rendszer logikai adatfolyamok

A vezérlő adatfolyamban mozgatott metaadatok körét tekintve az alábbi fontosabb metaadat típusok használatosak

A forrás adatokhoz kapcsolódó metaadatok:

- Forrás séma
- Nyomtatási lehetőségek források
- Források tárolási formátum leírása
- URL cím
- Tulajdonosi viszonyok
- Adattartalom leírás
- Források UPDATE gyakorisága
- Hozzáférés korlátok
- Frissítési ütemezés
- Hozzáférési jogok
- Átemelési rutinok elérése
- Átemelési rutinok beállításai

A köztes tárterületen elvégzendő feladatokhoz kapcsolódó metaadatok listája:

- Köztes állományok specifikációja
- DW dimenziók és változók specifikációja
- Konverziós rutinok specifikációja
- Konverziós rutinok ütemezése
- Változó dimenziók kezelésének paraméterei

- Kulcs generálás paraméterei
- Adat tisztítási paraméterek
- Adatelem leképzési szabályok
- Adat transzformációs szabályok
- Aggregációk definíciója
- Aggregációs, betöltési naplók
- Adat feldolgozási naplók
- Védelmi adatok

A DW rendszeren belüli, az adattároláshoz kapcsolódó metaadatok:

- DBMS rendszer táblák
- DBMS partíciók
- Indexek
- Fizikai tárolási paraméterek
- DBMS védelmi
- View definíciók
- Tárolt eljárások

A megjelenítési réteghez kapcsolódó leíró adatok:

- Jelentések, lekérdezések definíciója
- Lekérdezési segédeszközök elérése
- Lekérdezési segédeszközök paraméterezése
- Nyomtató paraméterezés
- Védelmi adatok
- Felhasználói beállítások
- Adatelérési útvonalak
- Felhasználási statisztikák, naplók

Az előzőekben vázolt jellemzések az adattárházról egy globális, áttekintő képet adtak, mely segítségével jobban megérthető a DW rendszerek szerepe, működési alapelve. A DW rendszer tényeleges használata, a benne rejlő lehetőségek pontosabb megismerése előtt azonban képet kell kapnunk a rendszer részletesebb működéséről, a rendszer használatának elemeiről. A mélyebb megismeréshez ezért most áttekintést következik a DW főbb moduljaihoz kapcsolódóan. Elsőként a DW rendszer magját alkotó adattárolási modult tekintjük át, mivel az adatok tárolási formátuma, az alkalmazott adatmodell jellege a rendszer többi elemére is kihat. Így a külső komponensek működése is csak akkor érthető meg pontosabban, ha ismerjük az azt meghatározó adatmodell, adattárolási rendszer működése.

## Multidimenzionális adatmodell

Az adattártárolás megvalósításának elemzése kapcsán gondoljunk most újra vissza a hagyományos, relációs adatmodell alkalmazási korlátjaira vonatkozó megjegyezéseinkre. Mint említettük, a relációs adatmodell igen rugalmas és hatékony egy programozó szemével nézve, de egy normál felhasználó személetmódját véve alapul, már nem állítható, hogy a relációs adatmodell a saját szemszögéből nézve is rugalmas és hatékony. A relációs modell legnagyobb nehézsége ezen a téren az, hogy az adatokat, információkat több normalizált adatszigetre, táblázatra bontja szét. Vegyünk például egy vállalati termelési igazgatóját, akit például a termelés alakulása érdekel a keleti régióra vonatkozóan az elmúlt három hónapra

vonatkoztatva. Nos egy relációs tervezésben jártas személy számára rögtön látható, hogy a relációs adatmodellt véve alapul, itt legalább három táblázatot fog tartalmazni az adatbázis szegmens, melyből kinyerhető a kívánt információ. Ez a három tábla a következő:

TELEP(tkod, nev, kozpont, regio,...)
TERMÉK(kod, megnevezes, egysegar,...)
TERMELES(termek, telep, datum, db, kategoria,...)

A valós információs rendszerek természetesen ennél sokkal több táblázatot is tartalmaznak, nem ritkák a több száz táblázatból álló rendszerek sem. Természetesen ekkor már külön szakember kell, aki képes megjegyezni ezen táblák nevét, jelentését, s kapcsolataikat. A mi példánkban viszont ezen három táblát alapul véve, a lekérdezéshez egy SQL szintaktikával megfogalmazott parancsot kell összeállítani. Ha csak egy egyszerű listával is megelégszünk, mely az egyes termékek napi forgalmát adja meg a megadott régió és időkorláton belül, akkor egy az alábbi parancssorokhoz hasonló utasításokat kell kiadnunk:

```
CREATE VIEW v1 AS SELECT termek, datum, sum(db) as odb FROM termeles
WHERE datum BETWEEN sysdate() AND sysdate() – 90 GROUP BY termek, datum;
```

```
SELECT b.megnevezes, c.odb, b.egysegar*c.odb as ertek, c.datum FROM Telep a,
termek b, v1 c WHERE a.tkod = c.telep AND c.termek = b.kod AND a.regio =
“Kelet” ORDER BY megnevezes, datum ;
```

Ha azonban nemcsak egy sima lista a kívánság, hanem egy táblázat, nevezetesen egy keresztreferencia táblázat, melynek sorai az egyes hónapok és oszlopai az egyes termékek, akkor még sokkal több energiát és ötletet kell befektetni a lekérdezés összeállításába, mivel az SQL nyelv maga nem rendelkezik közvetlen keresztreferencia készítő operátorral. Ezért ebben az esetben egy dinamikusan összeállított SQL utasításra van szükségünk. Egy ilyen dinamikus SQL parancs előállítására készíthetünk például egy SQL scriptet, azaz egy programszeletet, amely egy újabb SQL parancsot fog generálni. Ha nem is próbáljuk is ki most itt ezen utat, bizonyára elhihető, hogy ennek megalkotásához is szükség van megfelelő SQL gyakorlatra és szakértelemre. E példa alapján tehát látható, hogy az SQL nyelv egyszerűsége nem a hétköznapi értelemben vett egyszerűséget jelenti. A DW rendszerek általi kitűzött egyszerűség elérésére tehát egy újfajta adatmodellre van szükség.

Ha azonban elfogadjuk azt, hogy itt egy újfajta adatmodellre van szükség, akkor viszont már egy mélyebb reform is elkerülhetetlené válik. Ha ugyanis visszagondolunk a hatékonysági követelményekre, akkor látszik, hogy a DW rendszernek is elegendően gyorsnak kell lennie a versenyben maradáshoz. Ha azonban a régi relációs modellre igazított fizikai tárolási struktúra marad meg, akkor az új műveletek bizony elég gyenge teljesítménnyel fognak futni. Ennek oka, hogy minden adatbázis motor a saját adatmodelljére hangolt, az abban definiált operátorok hatékony végrehajtására optimalizáltak. Ha egy régi adatbázis végrehajtó programot egy új operátor készlet mellett működtetjük, akkor a kapott rendszer teljesítménye jelentősen elmarad az elérhető optimális teljesítménytől. Ezért DW adattárolást végző modul fizikai szintjét is át kell alakítani nemcsak a felhasználó oldalán megnyilvánuló adatmodell részt kell felújítani.

Az adatmodell részt illetően, a változtatások fő célja, hogy ne legyen az adatrendszer olyan sok kis részre szétdarabolva, a természetes szemléletmódban összetartozó adatelemek egy

helyen legyenek elérhetők. Az ezen új irányelveknek kielégítésére kidolgozott adatmodellt nevezik multidimenzionális adatmodellnek.

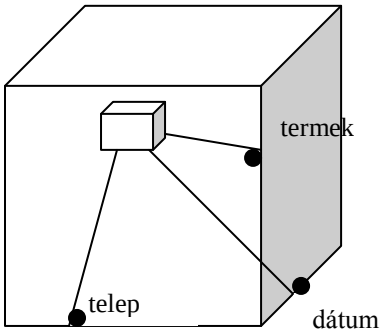
A *multidimenzionális adatmodellben* (*MD,* multi dimensional data model) a multidimenzionalitás arra utal, hogy itt az elemi adatokat nemcsak egy kulcs függvényében lehet elérni, hanem több kulcstól való függése is nyilvántartott az adatbázisban. Az egyes kulcsok mint dimenziók szerepelnek az adatelemek elérésekor. Ha például veszünk egy autó adatokat nyilvántartó rendszert, akkor abban egy tábla szolgál az autók alapadatinak nyilvántartására. Ezen táblában egy sor egy konkrét autó adatait írja le. Az oszlopok az egyes autó tulajdonságokhoz tartoznak.

| Rsz | Tipus | Ar | Szin | Év |
|---|---|---|---|---|
| R2 | Fiat | 1666 | Kék | 1987 |
| R6 | Opel | 2162 | Zöld | 1999 |
| R3 | Skoda | 3182 | Piros | 2001 |

8. ábra, az autó reláció táblázata

Az adatok keresésekor az adatbázis kezelő, az azonosító szerepet betöltő kulcsmező alapján fogja megkeresni az egyes autó rekordokat. Vagyis az autó rekordok egyetlen dimenzió mentén, lineárisan rendezettek. A példában a kulcs, vagyis a dimenzió a rendszám mező. A relációs adatbázis több, egy dimenziós táblázatból áll.

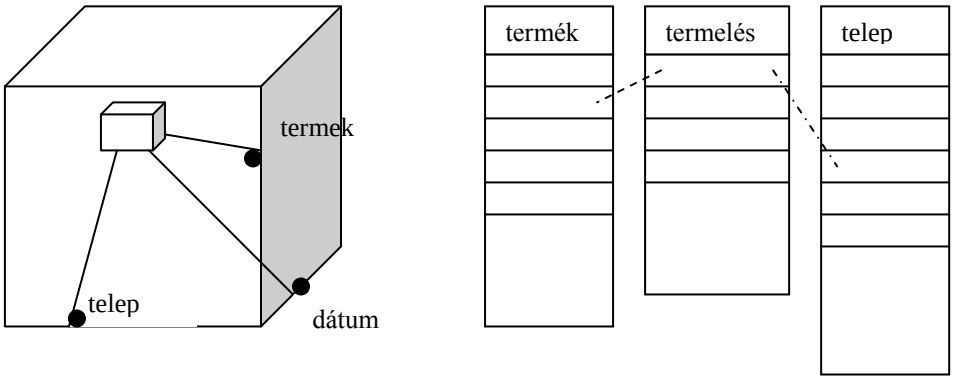
Az MD modellben viszont az egyes leíró adatok nem egy tengely, egy dimenzió mentén helyezkednek el, hanem egy több dimenziós térben. Az adatelemek ábrázolására ekkor egy kockát szoktak alkalmazni, amit adatkockának neveznek. Ha például egy három dimenziós esetet veszünk, akkor az adatkocka a következőképpen szemléltethető.




9. ábra, az adatkocka felépítése

Az egyes dimenziók itt az egyes kulcsokhoz tartoznak, csak most itt több kulcs is tartozhat egyetlen elemi adatstruktúrához. Ha például vesszük a korábban említett TELEP, TERMEK, TERMELES példát, amelyben a termelési adatokat tekintjük alapadatoknak, akkor a telep, a dátum, a termék fogalmak lehetnek az egyes kulcsok, azaz dimenziók. Így a termelési adatokat a telep, a termék és a dátum függvényében adhatjuk meg, kérdezhetjük le. Megadva például egy terméket, egy időintervallumot és egy telephelyet, a DW rendszer automatikusan rendelkezésünkre bocsátja az ezen dimenzió értékekhez tartozó termelési adatokat. Természetesen, mint majd az később látni fogjuk, a rendszer azt is megengedi, hogy ne kelljen megadnunk minden dimenzió értékét, elegendő csak bizonyos dimenziók leszűkítése, a többi dimenzió mentén nem végez semmilyen szűkítést. A rendszer további előnyös

szolgáltatása, hogy a felhasználó természetes módon, a szemléletéhez közel álló módon kaphat összesített adatokat is az egyedi adatértékek mellett. S ami igen fontos ebben az adatmodellben a felhasználó együtt láthatja az összes összetartozó adatot, egy adatkocka összefogja az összes adatot, ami a relációs modellben darabokban volt letárolva. A következő ábra a mintára mutatja be a két adatmodell különbségét.




10. ábra, az adatkocka és relációs modell összevetése

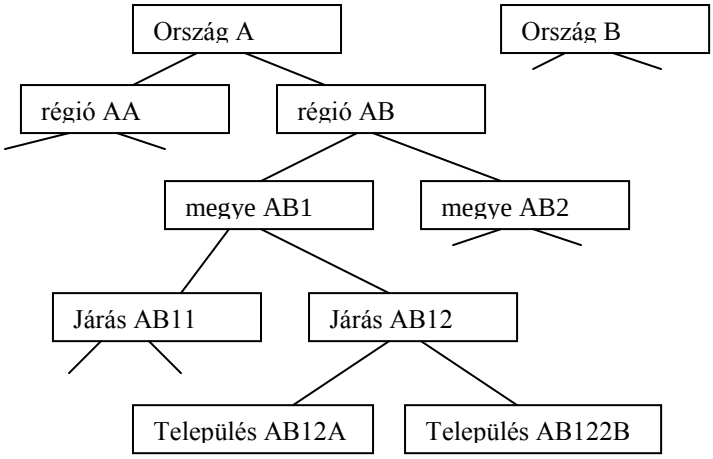
Mint a fenti példából is látható, ugyanazt az információ tartalmat képeztük le mind a relációs mind az MD modellre információ veszteség nélkül. Vagyis az MD modell nem a tárolt információ tartalmának a terjedelmében nyújt többet a relációs modellnél, nem jelent egy erősebb információ tárolási struktúrát nála, hanem az információ tárolásának a módjában van lényeges eltérés a két modell között. Míg a relációs modell alapvetően szétdaraboltan tárolja az adatokat, az MD modell egy adatkockába összehozza a logikailag összetartozó adatokat. A teljességhez még az is hozzátartozik, hogy azért a relációs modellben is meg lehet oldani, hogy a mintában megadott három tábla adatait egyetlen táblába hozzuk össze, azonban ez a táblázat semmiképpen sem egyenérték a megadott adatkockával, hiszen ez a táblázat nem normalizált a relációs fogalmak szerint, így kezelése problémákkal tűzdelt, s másrészt továbbra is csak a relációs modell operátorait használhatunk, s nem élhetünk a rugalmasabb MD modellbeli lehetőségekkel.

Az MD modell strukturális részének alapját képező adatkocka egyes szerkezeti elemeinek elnevezésére külön fogalmak alakultak ki az adatkezelésben. Az adatkocka legfontosabb elemei a következők:

- *Tény*: amit tárolunk az adatkockában, aminek az értékeit vizsgáljuk. Ilyen tény lehet például a rendelési érték vagy a termelési érték. Rendszerint a tény is több elemei tagból állhat. Például a rendelési adatnál a mennyiségi egység, a mennyiség, a tömeg alkothatják a tény elem részeit. Az adatkezelési lehetőségeket alapvetően befolyásolja, hogy a tényben tolt adatok aggregálhatók-e, azaz összeadhatók-e vagy sem. Ez alapján a következő ténytípusokat szokás megkülönböztetni:
    - Additive: összeadható, mint például a darabszám
    - Semi-additive: csak bizonyos részhalmazok elemei adhatók össze
    - Non-additive: nem összeadható, mint például a szín érték
- *Dimenzió*: a kulcsok, amiknek a függvényében érhetjük el az egyes tényadatokat. A rendelésnél lehet a vevő, a dátum vagy a termék ilyen dimenzió.
- *Dimenzió érték*: Az egyes dimenzió tengelyek több dimenzió értékre felosztottak. Az egyes dimenzió értékek az adott kulcshoz tartozó lehetséges értékek halmazát

jelöli. Így például dátum esetén az egyes napok, hónapok, évek vagy órák lehetnek a dimenzió értékei.

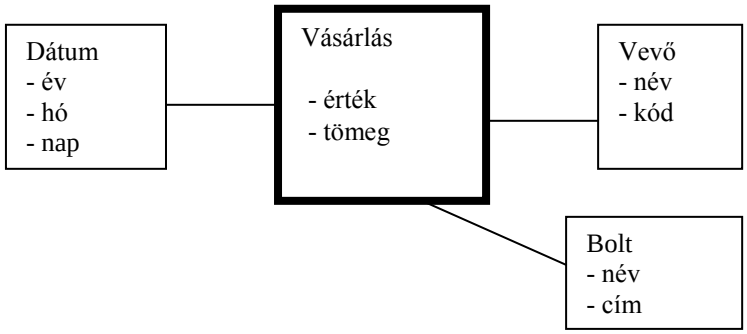
- *Tulajdonság*: az egyes dimenzió értékek a tényekhez hasonlóan összetett struktúrával is rendelkezhetnek. Ebben az esetben a egyes struktúra elemeket nevezik tulajdonságoknak. Egy termék dimenziót példaként véve a termék megnevezése, egységára, kódja lehetnek a tulajdonságai.
- *Adatcella*: az adatkockában a tény adatok az egyes dimenzió értékekhez kötöttek. Így az adatkockát úgy képzelhetjük el, mint több, nagy kockába rendezett elemi kocka, vagyis adatcella rendszere. Egy adatcella egy konkrét dimenzió érték n-eshez tartozik, ahol n a dimenziók darabszáma. Minden dimenzió érték n-es kölcsönösen egyértelműen kijelöl egy cellát az adatkockában.
- *Dimenzió hierarchia*: Egyes adatkockáknál előfordul, hogy a dimenziókra különböző finomság szerint rendezetten van szükség. Így például az idő dimenzió esetén ugyanazt a tényhalmazt egyszer év, másszor hónap felbontásban kell elemezni. Ehhez egyik lehetőség, hogy logikailag két különböző adatkockát hozunk létre a két különböző idődimenzióval. Ez azonban redundáns elemeket hoz be a rendszerbe, ezért a dimenzió hierarchia szolgáltat megoldást erre a problémára. A *dimenzió hierarchia* azt jelenti, hogy egyazon dimenzió tengely mentén több különböző dimenzió értékhalmaz helyezkedik el, úgy hogy az egyik felbontás minden eleméhez egyértelműen hozzárendelhető a másik felbontás egyetlen értéke. Így például a dátumnál maradva, a hónap felbontási részben minden hónap egyetlen év értékhez tartozik. A következő ábra a telephely dimenzióhoz kapcsolható hierarchiát mutat be.




11. ábra, dimenzió hierarchia

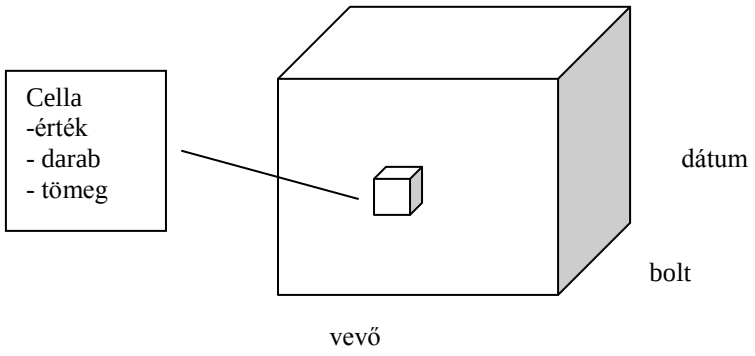
Az MD alkalmazások tervezésénél a tervezés első fázisában az adatkockák meghatározása történik. Az elkészült tervet rendszerint többen és többször is elemzik, mielőtt megvalósításra kerülne. Ehhez a tervet úgy kell leírni, hogy minél szélesebb körben alkalmazható, értelmezhető legyen, tehát egy szemantikai adatmodellt célszerű készíteni. A szemantikai adatmodellek emberközeli, grafikus jelölésrendszert használva adják meg a modell legfontosabb lényegi elemeit. A relációs modellben az egyed-kapcsolat (ER) modell volt ilyen általánosan elterjedt szemantikai adatmodell. Az MD rendszerben kétfajta grafikus jelölésrendszer terjedt el az adatrendszer struktúrájának szemléletes, lényegi elemeket tartalmazó megadására: egyik a csillag modell, a másik pedig a hópehely modell.

A *csillag modellben* az elkészült terv grafikus leírásának alakja emlékeztet a csillag formátumra. A modell célja az adatkocka szerkezetének megadása. Az ábrázolás középen áll az adatcella, a tény elem egy téglalappal szemléltetve. A tény leírásnál az egyes elemek felsorolása is szerepel. Ehhez az egyes dimenziók, a dimenzió értékek szerkezetének megadása kapcsolódnak csillagszerűen. A dimenzióknál is megadjuk a dimenziót leíró tulajdonságokat. A dimenziók ábrázolása is téglalappal történik. Az egyes kapcsolódó dimenzió és tény elemeket vonallal kötik össze.




12. ábra, Csillag modell a vásárlásokra

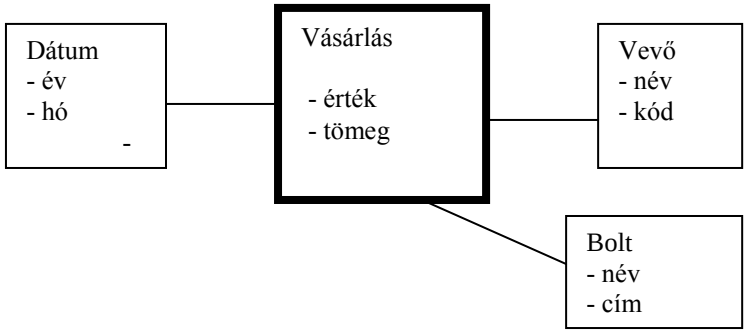
Az ábrában vásárlásokat leíró adatkocka sémáját adtuk meg. A középen elhelyezkedő tény kockában a vásárlást jellemző mennyiségek szerepelnek, mint az érték, tömege vagy darabszám. Ezek azok a mennyiségek, amelyek meghatározóak a rendelések jellemzése során, melyek önmagában a vásárlást jellemzik. Így most például értelmes lekérdezés lehet a vásárlások összárára vonatkozó információigény. A központi kockához a széleken azok a mennyiségek kapcsolódnak, melyek dimenzióként szerepelnek. Minden vásárláshoz egyértelműen kapcsolódik mindegyik dimenzió egy-egy konkrét érte. A mintában dimenzióként szerepel a vásárlás dátuma, a bolt, s a vevő. A sémában az egyes dimenziók jellemzése is szerepel. A vevő esetében például a vevő neve, lakcíme és kora szerepel adatként. Ezek az adatok fognak szerepelni az adatbázisban is, s ezen adatokat lehet majd felhasználni az adatkocka műveletek során is. Így például a vásárlási összérték lekérdezésekor le lehet szűkíteni a figyelembe vett vásárlások körét azon vásárlásokra, melyekben a vevő egy megadott kor alatti.




13. ábra, Adatkocka a vásárlásokra

A fenti séma alapján egyértelműen elkészíthető egy adatkocka a konkrét kiválasztott DW rendszer leíró nyelvében is. A 13. ábra az előző sémának megfelelő adatkockát ábrázolja. A programozónak, aki felépíti a sémához tartozó fizikai adatrendszert, a fenti kocka szerkezetét még majd előbb át kell alakítania a DW rendszer által értelmezhető parancsokká, mielőtt tényelegesen létrejön a fizikai adattárház.

A csillag modell mellett használatos *hópehely modell* is a séma leírás geometriai alakjáról kapta a nevét. A hópehely modellnél a csillag modellel ellentétben megengedett a dimenzió hierarchia ábrázolása, alkalmazása is. Ezen különbségtől eltekintve a két modell ugyanazon elemekből épül fel. A dimenzió hierarchia esetén a sémában egy dimenziót jelképező téglalaphoz egy újabb dimenziót reprezentáló téglalap kapcsolódhat. Az egy láncra felfűzött dimenziók ugyanazon alapdimenzió különböző finomságú felbontását jelentik. A ténytáblához közelebb álló dimenziótag jelenti a finomabb felbontási szintet. Ha földrajzi helyiséget, pozíciót veszünk a dimenzió jelentésére, akkor a láncban a hely megjelölés különböző finomsági szintjei szerepelhetnek, olyanok mint például a település, járás, megye, régió vagy az ország mennyiségek. A következő ábra a vásárláshoz kapcsolódó hópehely sémát mutatja, melyben a dátum felbontásra került év, hó és nap szintekre, valamint a bolt dimenziónál is új szintek jelentek meg, mint a régió, város a korábbi bolt kijelölés mellett. Mint a példában is látható, a dimenzió hierarchia minden tagjánál megadhatjuk a kapcsolódó tulajdonság értékeket a szokásos jelölés módot használva.



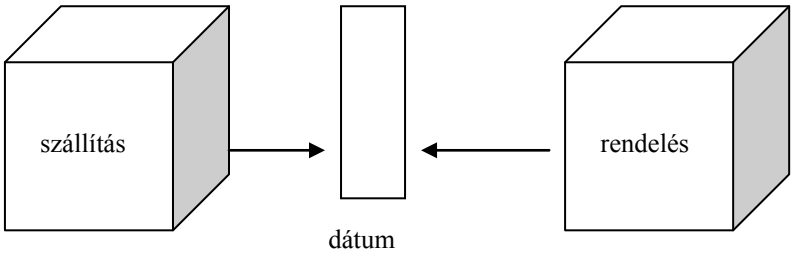

14. ábra, Hópehely modell a vásárlásokra

Az adattárház alkalmazások döntő többségében a modellezett problémakör nem annyira egyszerű, hogy egyetlen egy adatkockával leírható lenne. Hiszen a vizsgált területen több egymáshoz szorosabban vagy lazábban kapcsolódó tevékenységek folynak, melyek mindegyike egy önálló adatkockát igényel. Egy kereskedelmi vállalat esetében például külön adatkockák hozhatók létre az alábbi tevékenységekhez:

- eladások
- beszerzések, rendelések
- reklamációk
- beszállítások

A vállalatot leíró sémában ebben az esetben több adatkocka is szerepelni fog. Rendszerint mindegyik kockát külön szerepeltetik a sémában, nem hozzák össze őket egyetlen hálóba, mint azt a relációs modellhez igazodó EK modellben teszik. Több adatkocka alkalmazása esetén viszont gyakran előfordul, hogy több adatkocka is használja ugyanazt a dimenziót. Egy vevő dimenzió például szerepelhet a vásárlások valamint a reklamációk adatkockánál is. Ebben az esetben ugyan a vevő dimenzió többször is előfordul a séma rajzban, de a fizikai megvalósításnál rendszerint csak egyetlen egy fizikai vevő dimenzió adatsor fog letárolásra

kerülni az adattárházban, azaz egy dimenzió értékhalmaz megosztásra kerül több adatkocka között.




15. ábra, dimenzió megosztás

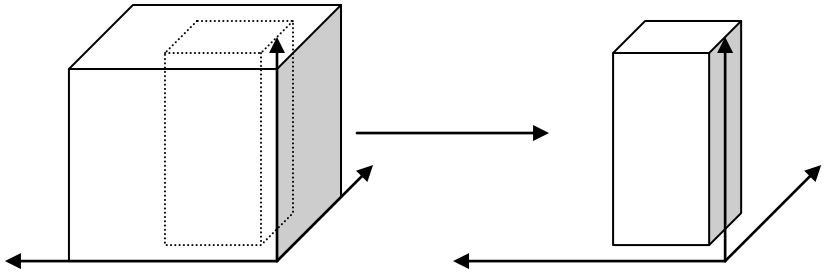
Az adatkockák szerkezetének áttekintése után most az adatkockához kapcsolódó műveleteket vesszük sorra, hiszen az adattárházak fő célja, hogy hasznos információt nyerjenek ki belőle a működése során. Az információ lekérdezéséhez viszont ismerni kell a rendelkezésre álló műveleteket, hogy milyen módon, milyen átalakításokkal kaphatjuk meg a nagy adatkockából a számunkra fontos adatrészt. A lehetséges adatkocka kezelő műveletek halmazából most a kocka tartalmának kinyerésére vonatkozó műveletekre koncentrálunk, hiszen az alkalmazások döntően információ lekérdezési szándékkal futnak.

Az egyik leggyakrabban használt művelet az adatkocka leszűkítése, vagyis amikor nem a teljes kockára, hanem annak csak egy szeletére van szükségünk. Ezt a műveletet nevezik szűkítésnek, vagy angolul “slice and dice” műveletnek. Ezzel a művelettel a teljes adatkocka tartalmát szelektálhatjuk, szűrhetjük egy általunk megadott szűkítési feltétel alapján. A szűkítés egyértelmű elvégzéséhez a következő paramétereket kell megadni:

- adatkocka azonosítása
- szelekciós feltételek

Ha például az említett vásárlások adatkockát vesszük alapul, akkor a teljes kocka minden vásárlás adatit tartalmazza, míg egy szeletelési művelet csak a feltételnek eleget tevő vásárlások adatait foglalja magába. Egy ilyen szűrési feltétel lehet például az, hogy a termék legyen labda, azaz csak a labda termékre vonatkozó vásárlások adatait fogjuk visszakapni eredményül.

A szűkítési művelet eredménye tehát egy újabb adatkocka lesz, amely a kiinduló adatkocka értékinek egy részhalmazát tartalmazza. Egy szűkítési műlettel lehet egy dimenzió de akár több dimenzió szerint is szűkíteni. A következő ábra egy több dimenzió szerinti szűkítési műveletet ábrázol a vásárlások adatkockára, ahol egy összetett, több dimenziót is érintő feltételt adtunk meg. Az alkalmazott szelekciós feltétel: a termék legyen labda és a dátum pedig a nyári hónapokat ölelje át.



16. ábra, szelekciós művelet

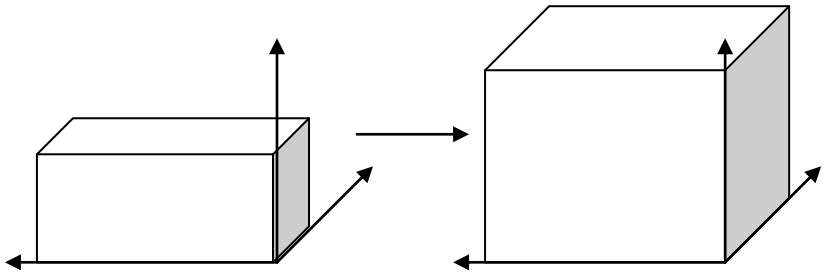
A szűkítés mellett, mellyel egy meglévő adathalmazból válogathatunk egy másik fontos adatkocka operátor a finomítás, vagy angolul a "drill down". Ez a műveletet nem egyszerűen szűkíti az adathalmazt, hanem kibővíti azt. A finomítás eredményeképpen több, részletesebb adatleírást fog tartalmazni az adatkocka, tehát itt is egy újabb adatkockát fogunk kapni eredményül. A finomítás során rendszerint

- egy újabb, részletesebb dimenzió hierarchiára térünk át vagy
- egy újabb attribútumot hozunk be a táblázatba az aktuális dimenzió szintjén

A finomítás hatására több információt, részletesebb szinten álló információt kaphat meg a felhasználó az adatkockában tárolt adatokról. A vásárlások adatkocka példánál maradva, ahol az adatok most éppen havi bontásban szerepelnek, a finomítással elérhető többek között az, hogy

- ne havi bontásban jelenjenek meg az adatok, hanem napi bontásban
- a vevőknél ne csak a neve szerepeljen hanem az életkora is megjelenjen

Az első műveletnél egy finomabb dimenzió hierarchia szintre térünk át, míg a második feladatnál a meglévő dimenzió hierarchia szinten maradunk, de egy újabb dimenzió tulajdonságot hozunk be az eredménybe. A finomítás műveletét szemlélteti a következő ábra.



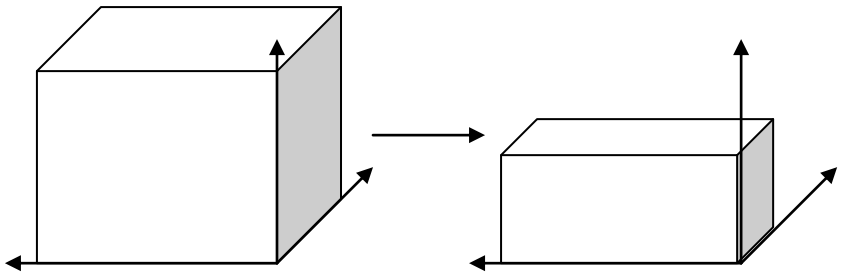
17. ábra, drill down művelet

A finomítást tehát akkor alkalmazhatjuk, ha egy részletesebb leírást szeretnénk kapni valamely adathalmazról. A részletesebb adatsor több információt hordoz, azonban a teljes adattárház a lehető legrészletesebben megjelenítve mégsem használható humán kiértékelésre, ugyanis véges, igen szűk azon információhalmaz nagysága, amit az agyunk egyszerre fel tud fogni és értelmezni is képes. Ezért a finomítást csak egy szűkebb adatmennyiségre érdemes végrehajtatni. Az stratégiai döntések meghozatalához viszont igen fontos az átfogó, áttekintő információk ismerete. Ehhez az adatokat már nem a lehető legnagyobb részletességben kell szemlélni, hanem megfelelő aggregáltsági, összegző szinten. Így tehát szükség lehet egy olyan adatkezelő műveletre is, melyben az adatokat az adott finomsági szintről egy durvább szintre visszük át.

A finomítási művelet ellentéte az összevonás művelete, vagyis a "drill up", amikor a jelentésből kiveszünk bizonyos mezőket. Ez a művelet magába foglalhatja a

- egy újabb, durvább dimenzió hierarchiára való visszatérést vagy
- egy dimenzió attribútum kivételét a táblázatból az aktuális dimenzió szintjén maradva

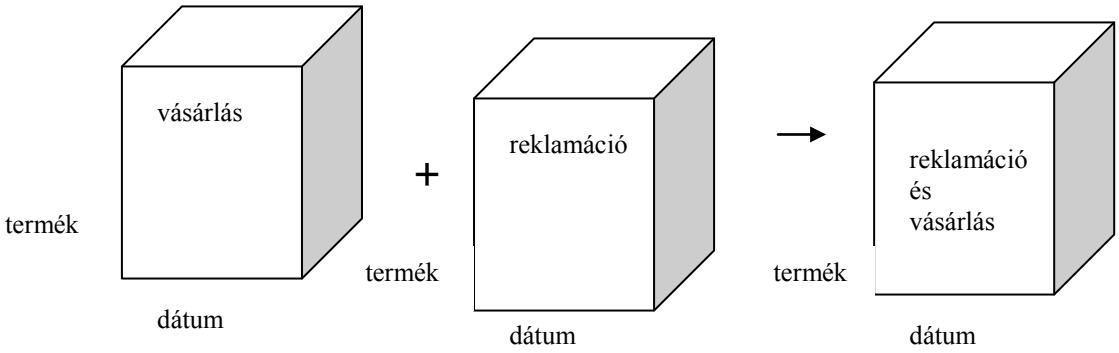
Az összevonás segítségével az egyes eddigi részletezett, adatelemek szintjén álló információkat összesítve láthatjuk. Ha például eddig heti bontásban és boltonként láthattuk a forgalom adatokat, akkor az összevonás révén át lehet térni havi idő bontásra, ahol az adatkockában az adott hónap dimenzióértéknél az oda tartozó napi adatok összesített értéke fog megjelenni.



17. ábra, drill up művelet

Hasonlóan elérhető az is, hogy a vevő dimenzió tengely mentén kevesebb tulajdonság szerepeljen az egyes vevők adatai között. Az összevonás is egy újabb adatkockát fog eredményezni, mint azt a következő ábra is szemlélteti.

Egy újabb adatkocka műveletet jelent az összekapcsolás művelete, aminek “drill across” a megfelelő angol elnevezése. Ez a művelet az előző műveletektől eltérően nem egyetlen egy adatkockával dolgozik, hanem két adatkockát vesz alapul, s az előálló eredmény adatkocka adatait ezen két adatkocka adataiból származtatja. Az összekapcsolás előfeltétele, a két adatkockának legyenek közös dimenziói. Egy ilyen esetet mutat be a következő ábra, melyben a vásárlás és a reklamációk tényadatai szerepelnek. A két adatkockának több közös dimenziója is van, mint például a dátum, és a termék.




18. ábra, összekapcsolás művelete

A közös dimenzióval rendelkező adatkockák összekapcsolhatók ezen közös dimenzió mentén. Vagyis az eredmény adatkockában együtt szerepelnek a két különböző táblából vett azon tényadatok, melyek ugyanazon dimenzió értékhez tartoznak a közös dimenzió mentén. Az előző példánál maradva, a dátum és a termék dimenziók szerint elvégezve az összekapcsolást, egy olyan eredmény táblát kapunk, melyben a közös dimenziók szerepelnek, tehát a dátum és a termék, a tényadatok pedig a két alapkocka tényadatait egyesíti, s így benne lesznek mind a vásárlások, mind a reklamációkra vonatkozó adatok. A felhasználó által kapott listák az összekapcsolás előtt és az összekapcsolás után a 19. és 20. ábrának megfelelően alakulnak.

Az összekapcsolás segítségével tehát össze lehet hozni a különböző adatkockákban elhelyezkedő adatokat, s ez a művelet az egyetlen, amely ezzel a tulajdonsággal rendelkezik.

Vásárlás

| Dátum | Vevő | Termék | Érték | Db |
|---|---|---|---|---|
| 10.02 | 45 | 786 | 155 | 2 |
| 10.13 | 67 | 678 | 65 | 1 |
| 10.16 | 87 | 78 | 83 | 3 |

Reklamáció

| Dátum | Termék | Hiba | Érték | ... |
|---|---|---|---|---|
| 10.12 | 625 | 77 | 144 | |
| 10.13 | 678 | 76 | 12 | |
| 10.16 | 78 | 99 | 1 | |

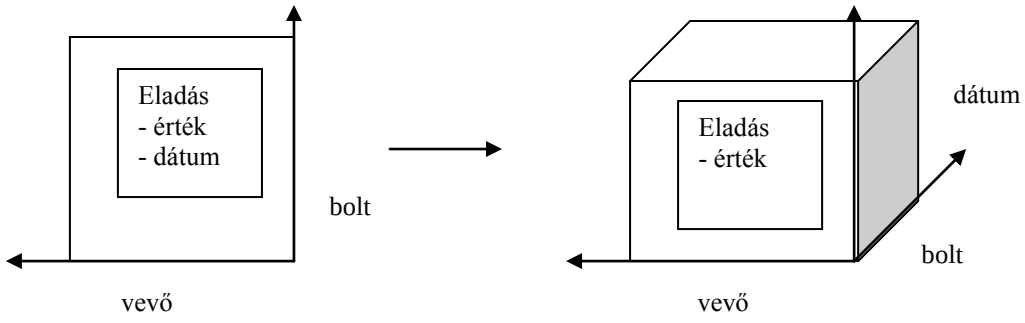
19. ábra, összekapcsolás előtti listák

Eredmény tábla

| Dátum | Vevő | Termék | Eérték | Db | Hiba | HÉrték |
|---|---|---|---|---|---|---|
| 10.02 | 45 | 786 | 155 | 2 | | |
| 10.13 | 67 | 678 | 65 | 1 | 76 | 12 |
| 10.16 | 87 | 78 | 83 | 3 | 99 | 1 |

20. ábra, összekapcsolás után listák

Egy újabb érdekes adatkocka műveletpárt jelentenek a bevonás és a kibontás műveletei. A kibontásnál (angolul "unfold") a tény struktúra megadott mezője átmegy dimenzióba. Vagyis az eredmény adatkocka egy újabb dimenzióval bővül, miközben a ténycella szerkezete egy mezővel csökken. A következő ábra ezt az átalakítást szemlélteti.




21. ábra, Unfold művelete

A kibontás segítségével meghatározhatjuk a vizsgált tényadatnak az egyik korábbi mezőjétől való függését. Ha például a vásárlások adatkockában a ténycella az alábbi mezőket tartalmazza:

- érték
- szín
- darab

illetve az alábbi dimenziók szerepelnek:

- dátum
- termék
- bolt

akkor a adatkockából a fenti értékeknek a dimenzióktól való függése vizsgálható, de az egyes mezők közötti kapcsolat nem jeleníthető meg. A kibontás segítségével elérhető viszont, hogy a szín mező kikerüljön a dimenziók közé, s így az eredmény adatkockában a mezőlista új alakja:

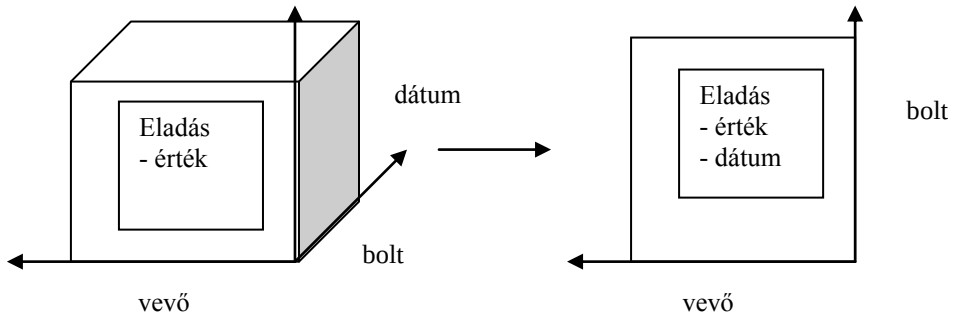
- érték
- darab

lesz, míg a új dimenziólista a

- dátum
- termék
- bolt
- szín

dimenziókat tartalmazza. Ezen eredmény adatkockában viszont már lekérdezhető a forgalom adatoknak a színtől való függése is a dátum, termék és bolt dimenzió függések mellett.

A kibontás műveletének párja a bevonás, vagy angolul a "fold" operátora. Ebben az esetben a dimenziók száma csökken, s a megszüntetett dimenzió értékei bekerülnek a ténycellába mezőként. Erre akkor kerülhet sor, ha valamely dimenzió átmenetileg érdektelenné válik, a tőle való függőségi viszonyt nem fogják vizsgálni a közeljövőben.




22. ábra, Fold művelete

Az előzőleg kapott adatkockát a

- dátum
- termék
- bolt
- szín

dimenziókkal egy bevonás művelettel lehet újra visszaalakítani az eredeti alakra. Ekkor a szín elemet kivesszük a dimenziók közül, s áttesszük a ténycella mezői közé. A művelet hatására eggyel csökken a dimenziók száma, s eggyel nő a ténycella mezőinek a darabszáma.

Az utolsó bemutatandó művelet egy nem az adatkockára, hanem a kapott listára vonatkozó átalakítást szemléltet. Hatására a lista szerkezete alapvetően megváltozik. Ez a művelet a pivotálás, amelynek során a listában korábban szereplő értékekből oszlopok lesznek. Ha például vesszük a vásárlások adatkockát, akkor az egyes vásárlások adatai az alábbi listával írható le

| Dátum | Vevő | Termék | EÉrték | Db | Hiba | HÉrték |
|---|---|---|---|---|---|---|
| 10.02 | 45 | 786 | 155 | 2 | | |
| 10.13 | 67 | 678 | 65 | 1 | 76 | 12 |
| 10.16 | 45 | 781 | 83 | 3 | 99 | 1 |
| 10.21 | 45 | 786 | 236 | 3 | | |
| 11.10 | 67 | 786 | 287 | 1 | | |

23. ábra, Induló minta tábla

Ahol az egyes sorok az egyes celláknak felelnek meg, s az oszlopok a tény illetve a kapcsolódó dimenzió értékeknek felelnek meg.

Ha a felhasználó nem az egyes konkrét vásárlásokra kíváncsi, hanem az azokból képzett összevont értékekre, melyek az előző lista átalakításával határozhatók meg az értékek oszlopként való felvételével, akkor a pivotálás művelete alkalmazható. Az előző listára vonatkozóan igényelhető például az, hogy a tulajdonságként szereplő területkódok és színkódok oszlopként, sorként szerepeljenek, azaz külön oszlopba vagy sorba kerüljenek az egyes területkódok és színkódok. Ekkor ezen sorok, oszlopok celláiban az adott terület- és színkódhoz tartozó összesített érték fog megjelenni. A kapott eredménylistát szemlélteti a következő ábra.

| | Vevő | 45 | 67 | ... | ... | ... |
|---|---|---|---|---|---|---|
| Termék | | | | | | |
| 678 | | 0 | 65 | ... | ... | ... |
| 781 | | 83 | 0 | ... | ... | ... |
| 786 | | 391 | 287 | ... | ... | ... |
| .... | | | | | | |

24. ábra, Pivotált tábla az értékekre

Maga a pivotálás, mivel csak megjelenítési lista alakját formálja át, s nem érinti az alatta lévő adatkockát, csak kiegészítő jelleggel szerepel az adatkocka műveletek között.

Az előzőekben vett műveletek nemcsak önállóan szerepelhetnek, hanem láncoltan is, vagyis az egyik lépés eredmény adatkockáján egy újabb műveletet hajtunk végre, s az így kapott újabb adatkocka egy esetleges újabb, soron következő művelettel tovább alakítható. A fenti lépések megfelelő láncolásával a felhasználó a kiinduló adatkockákból előállíthatja az elemzés során szükségessé váló adatlistákat. A fenti műveletek előnye az, hogy közel áll a hétköznapi gondolkodásmódhoz, nem igényel bonyolult paraméterezést, s mindegyik önmagában is használható és szükség esetén könnyen láncolható is.

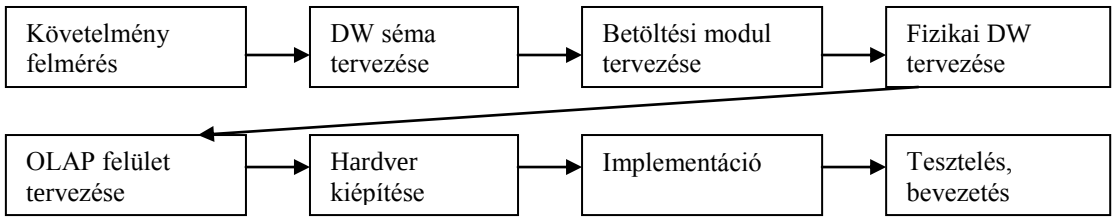
Egy ide vonatkozó záró példaként vegyük azt a műveletsort, amikor két adatkocka adott, a vásárlások s a reklamációk. A felhasználó előbb összekapcsolja a két adatkockát közös dimenziók mentén, majd összevonást végez, hogy lássa a globális adatokat. Ebben egyes területre vonatkozó adatok feltűnnek neki, ezért a kockát szűkíti a kijelölt területre. Ezt követően szeretné megismerni az ide vonatkozó részletesebb adatokat is, ezért egy finomítást hajt végre. A kapcsolatok pontosabb megismeréséhez a ténycellából átvisz egy mezőt a kibontás segítségével dimenzióba, egy újabb adatcellát kapva eredményül. Az így kapott adatkockát listában megjelenítve, a pivotálással több különböző szempont szerinti csoport összesítéseket is igényelhet az elemzés teljessé tételére.

## Adattárház rendszerek tervezése

Az adattárház rendszerek bevezetésének és sikeres működtetésének elengedhetetlen feltétele az alapos és rendszerezett tervezés és projekt irányítás. Mint minden informatikai projektnél itt is alapvető fontosságú, hogy a projektet nem az informatikai eszközök, szoftverek beszerzésével vagy kidolgozásával kell kezdeni, hanem előbb meg kell alapozni a projekt keretét, megadva a lehetőségeket, a célkitűzéseket, a szükséges tennivalókat és ütemezésüket,

s a használat keretfeltételeit. A legjobb eszközöket is beszerezve az eredmény igencsak kétségessé válik, ha a vállalatnál nem teremtődik meg a feltétele a DW rendszer hatékony működésének. Az adattárházak bevezetése nemcsak informatikai kérdés, s talán mondhatjuk azt, hogy elsősorban nem informatikai kérdés. A DW rendszerek alkalmazásánál igen fontos, hogy a vállalat működési struktúrája, az üzleti folyamatok menete is illeszkedjen az új eszközökhöz, ki tudja használni az abban rejlő lehetőségeket.

Az alapos és rendszerezett előkészítés érdekében érdemes áttekinteni, hogy milyen lépésekre is lesz szükség a projekt futása során. A projekt teljes életciklusát tekintve több fontos lépés is definiálható. A következőkben az []-ben megadott modellre épített, de azt bizonyos elemekkel kibővített rendszermodellt fogjuk áttekinteni. A projekt főbb lépéseit a. 25. ábra szemlélteti.




25. ábra, Projekt lépései

A teljes projektet lefedő tevékenységi rendszerben az alábbi funkció modulokat célszerű megkülönböztetni:

- projekt tervezési modul:
- üzleti követelmények elemzési modulja:
- adattárház tervezési modul:
- alkalmazás tervezési modul:
- technológiai rendszer tervezési modul:
- alkalmazás előkészítési modul:
- információs technológiai rendszer tervezése:
- IT eszközök, termékek beszerzése, installálása:
- Adatkocka séma modellezés
- Adatkocka fizikai modell:
- Adatbetöltési, tisztítási modul:
- Alkalmazások követelményanalízise:
- Alkalmazások kifejlesztése:
- Szervezet felkészítése:
- Tesztelés
- Bevezetés
- Karbatartás
- Projekt menedzsment:

A projekt tervezési modulban kell megadni a DW rendszer bevezetése által elérhető előnyöket, s a bevezetés célját. Pontosan definiálni kell a várható eredményeket, megadva azok pénzügyi oldalait is. A terv ezen kívül tartalmazza a megvalósíthatóságra vonatkozó indoklásokat is. A projektre vonatkozólag szerepelni kell még a projekthez tartozó feladatokat és azok ütemezését, költségét, s a feladatok teljesítésének kritériumait. Ezen kívül meg kell még adni a projektben résztvevők, csoportok adatait és feladatait, az esetlegesen szükségessé

váló átképzések, betanítások, új munkaerő költségelemzésével együtt. Nem szabad elfeledkezni a projekt lezárásának pontosításáról sem.

Az üzleti követelmények elemzési moduljában a tervezőknek meg kell érteni a vállalatnál zajló működési folyamatokat, azok célját és keretfeltételeit. Meg kell találni azokat a pontokat ahol javításra van szükség, s elemezni kell, hogy a DW rendszer alkalmazása mennyiben és hogyan tudná növelni a folyamatok hatékonyságát. A feltárt beillesztési pontoknál meg kell ismerni a DW rendszerek illesztésének lehetőségeit, hogy mennyiben lehet hasznos a DW által nyújtott szolgáltatások az adott területen. Pontosítani kell a DW üzleti előnyei mellett azon kritériumokat is, melyet a DW alkalmazása támaszthat a vállalat üzleti, ügyviteli folyamataival szemben.

Az adattárház tervezési modul egy átfogó modul, melyben az adattárház tartalmára vonatkozó tervezési lépések foglalnak helyet. Ebben a részben a szükséges előismeretek köre az informatikai képzettség mellett a vállalatnál folyó üzleti, ügyviteli folyamatok ismeretére is kiterjed.

Az alkalmazás tervezési modulban az adattárházhoz kapcsolódó alkalmazási területek feltárása, az egyes alkalmazások funkcióinak és felhasználói körének a pontosítása hajtódik végre. Ez is egy több részlépésből álló modul, melyben szintén szükség van az informatikai képzettség mellett a vállalatnál folyó üzleti, ügyviteli folyamatok ismeretére is. Ebben a modulban igen fontos az alkalmazói oldal aktív közreműködése a funkciók, az alkalmazási körülmények specifikálásánál.

A technológia rendszer tervezésének moduljánál a DW rendszer informatikai hátterének a meghatározása, megvalósítása történik. Ebben a modulban elsődlegesen az informatikai háttér a lényeges, hiszen itt a működéshez szükséges hardver és szoftver háttért kell pontosítani és működésbe hozni.

Az alkalmazás előkészítési modul a DW rendszernek a vállalat ügyviteli folyamataiba való zavartalan integrálásának a folyamatát hivatott elősegíteni. Ennek során tudatosítani kell a leendő felhasználókkal az új rendszer előnyeit és követelményeit is. Gondoskodni kell a felhasználók esetlegesen szükségessé váló, időben végrehajtott átképzéséről is.

Az információs technológiai rendszer tervezése almodulban az informatikusoknak a DW rendszer tervezett működési paramétereinek és a meglévő informatikai infrastruktúra ismeretében pontosítani kell a szükségessé váló informatikai beruházásokat, átszervezéseket. Ennek során többek között ki kell térni az alábbi lépésekre is:

- meglévő hardver struktúra feltérképezése
- meglévő szoftver rendszerek számbavétele
- az IT eszközhasználat aktuális helyzetének elemzése
- a kívánt DW rendszer informatikai paramétereinek elemzése
- a új és a régi rendszer kapcsolatának meghatározása
- az egyes implementációs alternatívák kidolgozása, megadva azok funkcionális és költség vetületeit is
- az alternatívák közül a megvalósítandó kiválasztása

Az IT eszközök, termékek beszerzése, installálása előtt már eldőlt, hogy melyik IT konfiguráció változat fog megvalósításra kerülni. A beszerzésnél rendszerint több szállító ajánlatait is figyelembe szokták venni, élve a versenyeztetés lehetőségével is. A beszerzés

során tehát előbb pontosítani kell az eszközigény kiírást, ügyelve az esetleges kompatibilitási követelményekre is. Ezt követően értékelni kell a bejött ajánlatokat, s ki kell választani a győztes beszállítót. A szállítás során ellenőrizni kell a bejött termékeket, majd részt kell venni a termékek installációjában. A sikeres installáció után kerülhet sor a rendszer próbaüzemére, mely során az egyes komponensek kipróbálásra kerülnek különböző teszt programok és mintafeladatokon keresztül. A sikeres tesztelést követően kerülhet sor a rendszer átadására és a leendő felhasználóknak az esetlegesen szükségessé váló betanítására.

Az adatkocka séma modellezés rakja le az adattárház rendszer alapját. A modellezés során ki kell térni a következő elemekre:

- a DW rendszer által letárolandó információ tartalom meghatározása
- az adatok forrásának kijelölése
- az alkalmazási területek alapján az egyes adatkockák tématerületeinek megadása
- a tényadatok kijelölése
- a dimenziók kijelölése
- a tényadatok és a dimenziók tulajdonságainak meghatározása
- az osztottan használt dimenziók kiválasztása
- a várható lekérdezési műveletek és azok gyakoriságának elemzése, s ez alapján a rendszer optimalizálására vonatkozó lépések megadása a rendszertervben

Az adatkocka fizikai modelljénél létre kell hozni azt a műveletsort, amelyet az installálásra került DW kezelő értelmezni és végrehajtani tud. Ez a műveletsor alapján jön fizikailag is létre az adatrendszer az adattárházon belül. A fizikai adatrendszer megalkotása többek között az alábbi lépéseket fogja össze:

- DW rendszer parancsnyelvének megismerése
- A séma leírás átkonvertálása a DW nyelvre
- Objektum azonosítási konvenciók megalkotása
- A tulajdonságok adattípusainak megadása
- Kulcsok kijelölése
- Integritási szabályok meghatározása
- Indexek létrehozása
- Előaggregált értékek kijelölése
- Hozzáférési védelmi rendszer kiépítése
- Osztott elérési mechanizmusok beépítése
- Adatvesztés elleni mechanizmusok megadása

Az adatbetöltési, tisztítási modulnál az DW rendszer bemenő oldalának a specifikálása és létrehozása történik. A tapasztalatok szerint ez az egyik legidőigényesebb lépés a DW rendszer tervezése, megalkotása során. A tervezőnek ebben a fázisban a következő feladatok kell elvégeznie:

- pontosítani kell az egyes bemenő adatok forráshelyét
- meg kell határozni az egyes forrás adatok információ tartalmát
- meg kell ismerni az egyes forrás adatok tárolási formátumát
- meg kell ismerni az egyes forrásokhoz való adathozzáférési módozatokat
- ki kell jelölni egy közös átmeneti adatterületet a beolvasáshoz
- meg kell írni az adatokat a forrásokból az átmeneti területekre átvivő programokat
- meg kell oldani az átmeneti területen az egyes bejövő adatok integrálását
- gondoskodni kell a különböző forrásokból származó adatok közötti ellentmondások kezeléséről

- meg kell írni az átmenti tároló helyről az adattárházba bevivő programot
- gondoskodni kell az adatoknak a DW rendszerben már meglévő adatokhoz való illesztéséről, az integritási szabályok betartásáról
- gondoskodni kell az elavult adatok kiürítéséről

Az alkalmazások követelmény analízise során fel kell tárni a felhasználók igényeit, a szokványos alkalmazási körülményeket. Ennek során ki kell térni az alábbi főbb tevékenységi pontokra is:

- a tipikus felhasználói csoportok meghatározása
- az egyes felhasználói csoportok információ igényének összegyűjtése
- az alkalmazások futtatási körülményeinek meghatározása figyelembe véve a rendelkezésre álló hardver és szoftver adottságokat is
- az információkhoz kapcsolódó tevékenységek összegyűjtése
- az információk megjelenítési formátumának meghatározása
- képernyő tervek kidolgozása
- a védelmi igények felmérése
- az alkalmazások rendszertervének kidolgozása
- az adat-funkció mátrix meghatározása
- alkalmazási modulok kijelölése
- az összetartozó adatelemek csoportosítása
- adatbázis aktív elemek kidolgozása

Az alkalmazások kifejlesztése során a kidolgozott rendszerterve építve kell kódolni és tesztelni az egyes alkalmazási modulokat. Ebben a lépésben elsődlegesen a programozói szakemberekre van szükség, akik az elkészült rendszer elemeket vagy prototípusokat a felhasználóknak bemutatják visszacsatolás végett. A fejlesztéshez rendszerint egy fejlettebb alkalmazás fejlesztő eszközt szokás alkalmazni, amely meggyorsítja az alap prototípusok elkészítését, s támogatja a csoportos fejlesztő munkavégzést is. Ebben az esetben már félig kész megoldások is rendelkezésre állhatnak, melyek megfelelő paraméterezése a fejlesztő munkájának része. A kódolási fázis legfontosabb lépései:

- az alkalmazások moduljainak elkülönítése
- az egyes modulok adatrendszerének meghatározása
- adatbázis kapcsolat kiépítése, tesztelése
- elemi funkcionális egységek meghatározása
- a funkció egységek közötti kapcsolat definiálása
- a felhasználói felületek pontosítása
- az egyes vezérlő elemek grafikus paramétereinek meghatározása
- az egyes vezérlő elemek viselkedési paramétereinek meghatározása
- a képernyő elemek kódolása
- funkciók, rutinok kódolása
- jelentések pontosítása
- jelentés készítő részek programozása
- modulok, rutinok kapcsolatának ellenőrzése
- a menürendszer kidolgozása
- a HELP, súgó rendszer kifejlesztése
- dokumentációk elkészítése

A szervezet felkészítése során a leendő alkalmazási környezetet kell ellenőrizni, hogy meglegyenek a rendszer bevezetésének és hatékony alkalmazásának feltételei. Meg kell találni és pontosan be kell mutatni azon érdekeket, melyek a rendszer bevezetése mellett

szólnak. A felkészítés során el kell végezni azon szervezeti és ügyviteli változtatásokat, amelyek a rendszer működtetéséhez szükségesek. Ki kell dolgozni az új rendszerhez kapcsolódó hozzáférési, dokumentálási, ellenőrzési szabályokat az érintett vállalati részlegekre.

A tesztelés is egy igen hosszú szakasza lehet a program kidolgozásának, ugyanis a tesztelés folyamata során meg kell győződni az egyes modulok helyes és hibamentes működéséről. Ehhez célszerű alkalmazni valamilyen validációs és verifikációs segédeszközt is. A tesztelés több fázisban szokott megtörténni, kezdve a belső, még modul, rutin szintű teszteléstől a külső, a felhasználási helyen történő tesztelésig. Ehhez a rendszert a végső helyre telepítve mintaadatokkal futtatják az eseteleges hibás funkciók feltárása végett.

A bevezetésre a sikeres tesztelést követően kerül sor, mely során a felhasználó rendeltetés és üzemszerű használatra átveszi az elkészült szoftver terméket. Az ide vonatkozó fontosabb lépések:

- a rendszer installált és tesztelt moduljainak átadása
- az adattárház feltöltése éles adatokkal
- kapcsolat kiépítése az adattárházhoz
- segítségnyújtás a felhasználóknak a rendszer megismerésében
- szervezeti működési szabályok módosításának érvényesítése

A karbantartás a rendszer későbbi működése során esetlegesen fellépő hibák korrigálására szolgál. A karbantartásnak több szintje lehet, melyek a nyújtott szolgáltatások tekintetében térnek el egymástól. A legalacsonyabb szinten csak telefonos tanácsadást biztosít a fejlesztő cég, míg magasabb szinteken a hiba bejelentése után megadott időn belül a helyszíni tanácsadásra, beállítás módosítások elvégezésére is sor kerülhet.

## Projekt menedzsment

A projekt menedzsment szerepe a projekt sikeres végrehajtásának a biztosítása. A menedzsmentnek végig kell követnie a projekt teljes szakaszát, ennek megfelelően az alábbi három fő menedzsment szakaszt szokták megkülönböztetni:

- előkészítési szakasz
- tervezési szakasz
- végrehajtás ellenőrzési szakasz.

Az előkészítési szakaszban meg kell alapozni a projekt létjogosultságát és be kell mutatni megvalósíthatóságát. Ehhez az alábbi kulcs lépések megtételére van szükség:

- A vállalatvezetés egyértelmű támogatásának megnyerése. Be kell mutatni, hogy a DW rendszer bevezetése milyen anyagi, üzleti előnyöket fog hozni a vállalat számára. Ennek során alapozni lehet a már meglévő igényekre, melyek a hatékonyabb, gyorsabb információszerzésre és döntéshozatalra irányulnak, másrészt más vállalatokat példaként véve bemutatható, hogy a DW rendszerek bevezetése ott milyen eredményeket hozott. A konkrét, világos és megvalósítható célok kijelölésével kell a DW rendszerek alkalmazhatóságát alátámasztani.
- Meg kell bizonyosodni afelől, hogy a vállalat valóban alkalmas az új DW technológiák befogadására, amihez elemezni kell a vállalat jelenlegi helyzetét. Egy dinamikusan fejlődő cég, vagy egy a piaci konkurencia harc jellegét felismerő cég,

amely világosan látja, hogy a jövőbeli sikeres működéshez milyen, az információ feldolgozás hatékonyabbá tételére irányuló lépésekre van szükség, hamarabb lesz kész beruházásokat eszközölni a DW rendszerek bevezetésére. Ezzel szemben egy leépülő, a kiadásokat jelentősen lefaragó vállalatok esetében nehezen lehet támogatást találni egy olyan nagyobb horderejű beruházáshoz, mint a DW rendszerek, hiszen egy DW rendszer csak hosszabb távon, aktív üzleti politikát folytató egységek esetében fog megtérülni.

- Össze kell hozni a vállalaton belül az informatikai és a üzleti szakembereket, és meg kell győzni őket, hogy csak a közös munka, az együttműködés hozhat sikert egy DW szintű projekt esetében, hiszen a DW tervezése, implementálása és működtetése e két oldal hatékony együttműködésén alapszik
- Elemezni kell a vállalatnál jelenleg folyó információ feldolgozási, döntéshozó folyamatokat. A DW rendszer ugyanis alapvetően ezen területeken hozhat javulást, változást a vállalat életében. Ha a vállalat az eddigiekben is alkalmazott valamilyen döntéstámogató módszereket, ha hangsúlyt fektetett a felgyülemlett információk elemzésére, rendszerezésére, akkor sokkal nagyobb eséllyel lehet bevezetni egy DW rendszert is, hiszen ekkor már van hagyománya az analitikus jellegű információ feldolgozásnak és döntés előkészítésnek a cég életében. Ha viszont eddigiekben csak a kézi módszerek, a megérzésen alapuló döntési eljárások kaptak helyet, akkor nagyobb harc árán lehet csak jelentős támogatást találni az új rendszerek bevezetéséhez is.
- Meg kell vizsgálni még a DW projekt elindítása előtt, a DW megvalósíthatóság is. Ennek során elemezni kell, hogy vajon minden adat rendelkezésre áll-e olyan formában, amit a számítógépes feldolgozás igényel. Ha igen, akkor meg kell nézni az adatok összegyűjtésének folyamatát a költségek, s az aktualitási szint szempontjából. Másrészt elemezni kell, hogy a vállalat üzleti folyamatai mennyire egységesek, mennyire lehet majd használni a DW rendszerbe bevitt adatokat és szabályokat a különböző részlegekben. Ha nincs elfogadott egységes értelmezés még, akkor elemezni kell ezen egységesítési folyamatok megvalósíthatóságát és költségigényét is.

Az új induló DW projektek esetén, amikor még nincs a szervezet birtokában kellő tapasztalat, mindenképpen egy kisebb méretű feladattal érdemes kezdeni, amely nem a teljes vállalat, hanem annak csak egy részlegének információ igényét fogja kielégíteni. Egy ilyen induló DataMart jellegű alkalmazási projekt előnye a rövidebb, maximum egy éves átfutási idő, a korlátozottabb terjedelem. Az így elkészült rendszert valószínűleg sokkal kevesebb felhasználó fogja használni, mint egy nagy vállalati rendszert, ezért ez kedvezőbb feltételeket teremt az induló tapasztalat gyűjtéshez is.

A rendszer előzetes tervezése során egyik fontos lépés a költségoldalak elemezése is, megadva a rendszer kiépítési költségeit, s a várható előnyöket is. A költség oldal elemzésekor ügyelni kell arra, hogy ne maradjon egyetlen egy fontos költségkomponens sem, mint például a

- a hardver elemek beruházási költsége
- a szoftver elemek beszerzési költsége
- a jövőbeli várható karbantartási költségek
- esetleges belső fejlesztési munkák (szoftver, szervezési, stb.) költségei
- külső vállalkozásoknak kiadott bérmunkák, eszköz bérlések költségei
- a felhasználók és karbantartók, programozók oktatásának költségei
- a rendszer által megkívánt átszervezési költségek

- a jövőben várható bővítésekhez kapcsolódó tervezett költségek

Az előzőekben említett költség oldallal szemben a várható költségek állnak a mérleg másik oldalán. Ugyan a bevezető részben már említettünk több olyan területet is, melyen javulás várható a DW rendszer bevezetésével, a megerősítés végett még egyszer összefoglaljuk azokat a pontokat, melyeket számításba lehet venni a DW bevezetése mellett érvek összegyűjtésekor:

- Hatékonyabb döntési eszközrendszer kiépítése
- Az egyes stratégiai alternatívák gyorsabb kiértékelése, több információ felhasználása a döntések meghozatalában
- A külső folyamatok változási irányának gyorsabb felismerése
- A folyamatok mögötti okok hatékonyabb megkeresése
- Célirányosabb tevékenységek végrehajtása
- Hatékonyabb ügyfél, vevő kiszolgálási szint
- Új üzleti lehetőségek hatékonyabb felismerése
- Célirányosabb piacpolitika megvalósítása
- Hatékonyabb készletgazdálkodás
- Hatásosabb kedvezmények, akciók megvalósítása

Ugyan a fenti listát még nyugodtan lehetne folytatni, de már az eddigiekből is látható, hogy az említendő előnyök alapvetően a DW információs rendszerén alapuló hatékonyabb, célirányosabb vállalati tevékenységből, magatartásból származnak. A DW tehát egy eszköz, egy újszerű lehetőség, melyet tudni kell megfelelően használni, hogy a kívánt célokat elérhessük vele.

*Tervezési szakasz*

A DW bevezetésére irányuló terv sikeres elfogadása után neki lehet kezdeni a projekt részletes kidolgozásának, megtervezésének. A projekt menedzser feladatai közé tartozik ebben a fázisban a projektben zajló munka megszervezése, beleértve a munkacsoportok kijelölését és az elvégzendő munkák meghatározását, ütemezését. Ez a feladat nagy szervezői tapasztalatot igényel, hiszen egy DW projekt igen sok szakember és szakterület hatékony együttműködését igényli. Már az is jól jellemzi a feladat nagyságát, ha csak azt vesszük is sorra, hogy milyen szakemberekre van szükség az egyes feladatok elvégzésénél. A következő lista ezen szereplőket összegzi, megadva a projektben betöltött szerepkörüket is.

- Vállalati vezetés, akik engedélyezték a projekt beindítását. Ez a csoport egyrészt felelőséget visel a projekt sikeres lefutását illetően, másrészt a projekt pénzügyi támogatása is tőlük függ.
- Projekt szakmai vezetői csoport, akik a konkrét projekt megtervezését végzik és ügyelnek a projekt sikeres lefuttatására ügyelnek. Ennek a gárdának kell közvetíteni a projektben résztvevő többi csoport között is, s dönteniük kell a menetközben fellépő szakmai problémák megoldásáról is.
- Projekt pénzügyi vezetői csoport: a projekt pénzügyi forrásainak megteremtésén dolgoznak, s ügyelnek a projekt szabályos, törvényszerű pénzügyi lebonyolítására, elvégzik a pénzügyi oldalhoz kapcsolódó adminisztrációs feladatokat is
- Szakmai, ügyviteli rendszerelemző: az ide tartozó szakemberek a vállalatnál folyó ügyviteli folyamatok jó ismerői, akik meg tudják adni a DW rendszer működési

környezetének paramétereit. Ők fogják meghatározni és kidolgozni a DW rendszernek az vállalati ügyviteli folyamatokba való illesztését, meghatározva, hogy milyen változtatásokra van szükség az ügyviteli folyamatok terén a DW rendszer befogadásához.

- Informatikai adatmodellező: az ügyviteli információigények ismeretében az ide tartozó munkatársak elvégzik az adatrendszer adatmodellezését, a szemantikus modellektől kezdve az adatbázis adatmodellig. Eközben meghatározzák az adatrendszerhez tartozó normalizálási, integritási és hatékonysági elemeket is.
- Adattárház adminisztrátor: olyan informatikai szakemberekre van itt szükség akik mélyen ismerik az általános adatbázis kezelési elvek mellett a kiválasztott adattárház kezelő rendszer működését, használatát és paraméterezését is.
- Adatátemelés és betöltés tervező: az idetartozó szakértőknek ismerniük kell a DW rendszerhez tartozó különböző adatforrások logikai és fizikai felépítését, meg tudják oldani az adatbetöltés során felmerülő konvertálási, adattisztítási feladatokat, s tisztában vannak az itt alkalmazható technikákkal, segédeszközökkel is.
- Alkalmazás tervező: A felhasználók által igénybe vett alkalmazói modulok rendszertervének az elkészítését végzik. Ismerniük és alkalmazniuk kell a megfelelő tervezési módszertanokat. A felhasználókkal is kommunikálva meghatározzák a megjelenítendő információkat, azok előállításának és megjelenítésének módját
- Alkalmazás fejlesztő: az ide tartozó programozói csoportnak az elkészült rendszerterv alapján meg kell írni a programok kódját, s el kell végezni a programok tesztelését is Ezen munkához ismerni kell a megfelelő fejlesztő eszközök, 4GL rendszerek használatát is.
- Oktatók: szerepük kettős, egyrészt részt vehetnek a belső fejlesztő gárda szakmai képzésében, másrészt a felhasználók oktatását is végezhetik. E csoport tagjainak nemcsak ismerniük kell a rendszer működésének elveit, technikáját, hanem megfelelően át is kell tudni adni a szükséges információkat.
- Informatikai rendszergazda: ezen a csoport feladata, hogy biztosítsa az adattárház informatikai rendszerének zavartalan működését. Ehhez ismerni kell az alkalmazott operációs rendszer működése, kezelése mellett a telepített hálózati rendszert is. A csoportnak fel kell készülnie a működés közben fellépő informatikai problémák megoldására is.
- Hardver szakértő, szerviz szakember: A rendszer informatikai hátterén belül a hardver komponensek zavartalan működéséért felelős csoportra is szükség van a csapaton belül.
- Adatátemelési modul programozó: míg az előző informatikai szakemberek általános informatikai feladatot láttak el, az adatátemelés programozónak már adattárház specifikus feladatokat kell ellátnia. Mint már korábban említettük, a DW rendszerek fejlesztésekor az egyik legnagyobb volumenű tevékenység az adatoknak a már meglévő adatforrásokból történő átemelése az adattárházba. Ezen rész nehézségét elődlegesen az okozza, hogy itt teljesen egyedi, vállalat specifikus keretfeltételek vannak az alapadatok forrásait illetően.
- Adat adminisztrátor: Az adat adminisztrátor szerepkörben tevékenykedő szakembernek a működő adattárházon belül tárolt adatrendszer épségéről, helyességéről, s az adatkezelés hatékonyságáról kell gondoskodnia. Ehhez jól kell ismerni a DW rendszer szemantikai tartalmát, s az adattárház kezelő rendszer kezelő parancsaiban is jártasnak kel lennie. Feladatát az adattárház adminisztrátorral együttműködve végzi.

- DW minőségbiztosítási szakértő: Napjainkban igen fontos szereplővé lépett elő a csapaton belül a rendszer minőségi követelményeinek betartásáért felelős csoport. Feladatik közé tartozik többek között a minőség ellenőrzési stratégia kidolgozása, az ellenőrzési pontok meghatározása, s a minőségellenőrzési folyamat felügyelete.

A fejlesztő csoport összeállítása, megtervezése után kezdődhet el az adattárház tényleges fejlesztési munkálata. Mint minden komplex tervezési feladatot, ezt is a követelmények összegyűjtésével, rendszerezésével kell kezdeni. A következő részben ezen információ gyűjtési szakaszról szólunk részletesebben.

*Követelmény specifikáció*

Az, hogy milyen lesz az adattárház belső adatszerkezete, a hozzá kapcsolódó alkalmazások működése, elsődlegesen a feltárt követelmények határozzák meg. Az adatbázis tervezők, vagy a program fejlesztők amikor kidolgozzák az informatikai rendszer részleteit, a megkapott követelmény specifikációt tartják szem előtt, nekik olyan rendszert kell tervezni, amelyek kielégítik a megadott követelményeket. Ezen jelleg szem előtt tartása azért is rendkívül fontos, mert az elkészült informatikai rendszerek későbbi módosítása igen költséges lehet. Nagy problémát fog jelenteni a későbbiekben, ha egy fontos szempont kimaradt a követelmények feltárásakor, mint például az, hogy a vevőnek a nevénél egyes helyeken a teljes névre szükség van, máshol viszont ismerni kell a családnevet, előtagot is. Ha a fejlesztő ezen követelményeket nem ismerve az informatikailag egyszerűbb megoldást választja, a példához kapcsolódóán egyben kezeli a teljes nevet, akkor csak a teljes rendszer vagy egyes modulok elkészülte után derül ki a tévedés. Mivel az információs rendszerekben sok elem bonyolult kapcsolatrendszerben egymásra épül, egy alsóbb szinten elhelyezkedő építőkocka kicserélése csak a felette álló rendszer esetleg teljes átdolgozásával lehetséges, hiszen a nem kellően átgondolt módosítások nagyobb károkat okozhatnak fel nem ismert mellékhatásaikkal, mint az alapprobléma jelentett. Ezért igen lényeges, mint a megrendelő, mind a fejlesztő oldaláról nézve, hogy pontosan tisztázottak legyenek már a fejlesztési munka megkezdése előtt, mint is kell pontosan tudnia a rendszernek, hogyan működjön.

A követelmény specifikáció során a felhasználók igényeit kell rendszerezni, s pontosítani kell a kidolgozandó rendszer funkcionalitását. Ezen elvárt funkcionalitási elemek fogják meghatározni a későbbi fejlesztési munkák lefolyását is. Így kihatással vannak többek között,

- az adattárház adatmodell kiépítésére,
- az alkalmazói modulok kidolgozására,
- az adatátemelési modul programozására,
- az informatikai konfiguráció kiválasztására,
- a későbbi fejlesztési lépések ütemezésére

A követelmény specifikáció során első lépés a felhasználói követelmények összegyűjtése. Ennek a lépésnek a legfontosabb megvalósításai a

- elbeszélgetések, találkozók
- gyűlések
- nyomtatott és elektronikus információ források tanulmányozása

A nyomtatott és elektronikus források tanulmányozása elsődlegesen az előkészítési szakaszban jellemző. Ekkor egy egyoldalú információ áramlásról van szó, melyben egy

általánosabb képet lehet kapni a megrendelő vállalat működéséről, a felépítési struktúrájáról, az esetleges távlati célokról, fejlesztési irányokról. Ezen anyagok segítségével az informatikus szakemberek is megismerkedhetnek a vállalat szakterületébe eső fogalmakkal, kifejezésekkel. Így a későbbi elbeszélgetések során nem fog gondot okozni az eltérő, szakma specifikus kifejezések megértése, használata.

A második fázist jelentő találkozók alkalmával egy-egy kisebb csoport tagjaival lehet információt cserélni. Ekkor közvetlenül is meg lehet ismerni a felhasználók tevékenységét, az ügyvitel menetét, s a munkamenethez kapcsolódó igények, elvárások, s az esetleges problémák is felszínre kerülnek. Másrészt ekkor a megrendelők is pontosabb képet kaphatnak a rendszer lehetséges szolgáltatásainak részleteiről. A sikeres információ gyűjtéshez azonban megfelelően elő kell készíteni a találkozókat, meg kell előre tervezni, hogy milyen információ elemekre van most szükség, s milyen formában lehet azt a leghatékonyabban megkapni.

A gyűlések egy nagyobb résztvevői létszámmal lezajló események, melyek elsődleges célja a nagyobb kört érintő kérdések megbeszélése, mint például

- a korábban felmerült általános problémák bemutatása
- az elért eredmények összefoglalása
- javaslatok összegyűjtése és összevetése

Az találkozók során igyekezni kell nyitott, a kérdezett személy területéhez szorosan kapcsolódó kérdéseket feltenni. Az alábbiakban, az [] alapján felsoroljuk, hogy melyek azok a tipikus kérdéskörök az egyes funkcióknál, amelyeknek célszerű belevenni a beszélgetésbe.

Ügyvezetők, igazgatók:

- a vállalat főbb fejlesztési céljai
- főbb teljesítmény mérőszámok
- legfontosabb aktuális problémák
- problémák lehetséges okai, hogyan lehetne megoldani őket
- hogyan kívánják hasznosítani a megnövekedő információ mennyiséget
- milyen információkat igényelne a döntési folyamatok támogatásához

Részleg vezetők

- a részleg főbb működési céljai
- főbb teljesítmény mérőszámok
- aktuális kihívások, feladatok
- teljesítmény értékelési módszerek
- termékek leírása, kategorizálása
- információ feldolgozás jelenlegi menete
- adatfeldolgozás jelenlegi menete
- adatforrások jellemzése
- milyen támogatásra lenne szüksége az adatok kiértékelésében,
- milyen információkat igényelne a döntési folyamatok támogatásához
- milyen jelentésekre van szükség

Informatikai szakemberek

- milyen információs rendszerek működnek már a vállalatnál
- meglévő rendszerek struktúrája, kapcsolata
- információ feldolgozás jelenlegi menete, technikája

- milyen jelentések készülnek
- az ad-hoc lekérdezési lehetőségek
- információs rendszerek minőségbiztosítási elemei
- adattárházzal szemben támasztott követelmények

Az eddigiek alapján érzékelhető, hogy az adattárház tervezésének folyamata, az implementációt megelőző rész önmagában is milyen összetett és sokrétű folyamat. Emellett azt sem szabad azonban elfelejteni, hogy a tervezés csak az induló szakasza a DW kidolgozása során. Természetesen az a szakasz az egyik legfontosabb szakasz, hiszen megfelelő megalapozás nélkül nem hozható létre hatékonyan működő információs rendszer. Hogy a teljes munkafolyamatról is megfelelő áttekintésünk legyen, a következő részben a tervezést követő lépéseket is befoglaló áttekintést adunk a DW rendszerek fejlesztési munkafázisairól.

*DW fejlesztési mátrix*

A tervezés folyamatában az első, már érintett szakasz a követelmény elemzéshez kapcsolódik, melyben az adattárházban letárolásra kerülő információ elemek és az igényelt műveletek köre kerül meghatározásra. A tervezés alapja az összegyűjtött felhasználói információ és funkció igények. Az elemzés során ugyan az igények irányítják a rendszerterv kialakítását, azonban az igényeket a meglévő keretfeltételekhez kell igazítani. A tervezés során ki kell dolgozni, hogy milyen módon lehet az igényelt információ elemeket a meglévő adatforrásokból átemelni, hogyan lehet a bejövő adatokat a kiépítendő DW rendszerbe integrálni, s elemezni kell, hogy a DW rendszer el tudja-e majd végezni az igényelt adatelemzési műveleteket, rendelkezésre állnak-e az igényelt DW funkciók. Ezen szempontok mellett a teljes körű elemzés a megvalósításhoz szükséges hardver, szoftver és szervezési igényekre is kitér.
Az itt felsorolt műveleteket a művelet jellege, tartalma alapján három fő csoportba sorolhatjuk.

1. DW struktúra tervezési lépések, tartalom megadás
2. DW-hez kötődő feldolgozási algoritmusok
3. DW megvalósítás hardver és szoftver háttere

Az első csoportba tartozik az a lépés, amikor az adattárházban letárolásra kerülő információ elemeket gyűjtjük össze, s elkészül a DW rendszer információ tartalom szinten történő megadása. A második csoport műveletei az előző lépésekben kidolgozott struktúra elemekhez kapcsolódnak. Ide tartozik többek között az adatoknak a különböző forrásokból való átemeléséhez tartozó logikai szintű leírása és az adatbázisba történő integrálás folyamata is. A harmadik csoport az implementációhoz kapcsolódó hardver és szoftver háttér elemzését öleli fel.

Az előzőekben felvázolt hármas csoportosítás, melyet most a tervezés első műveletihez kötöttünk, az adattárház rendszer kidolgozás többi fázisában is alkalmazhatónak bizonyul. A fejlesztés további eleminek számbavételekor azt a folyamatot kell végig gondolni, amikor az absztrakt, logikai szintű leírásból létrejön a működő, megadott hardver szoftver konfigurációban működő programrendszer. E folyamat részletes elemzése alapján a fejlesztés folyamatát az alábbi öt fontosabb lépcsőre szokás bontani:

- követelmény elemzés, megvalósíthatósági terv

- durva struktúra elemzés, tervezés
- részletes rendszerterv elkészítése
- implementáció
- tesztelés

Az egyes lépésekhez és művelet csoportokhoz tartozó elemi műveletek rendszerezésére, átfogó ábrázolására a legjobb megoldás egy táblázat, amelynek sorai az egyes fejlesztési lépcsőfokoknak, míg oszlopai az egyes művelet csoportoknak felelnek meg. Az így kialakuló táblázatot szokás fejlesztési mátrixnak is nevezni, amely összefoglalja a legfontosabb fejlesztési lépéseket, kijelölve a műveletek sorrendiségét, egymásra épülését is.

A követelmény elemzést követő durva tervezési fázisban a megvalósítandó adatrendszer meghatározása kezdődik el. Az adatrendszer leírására már nem egy emberközeli szöveges vagy grafikus jelölést rendszert alkalmazunk, hanem egy DW specifikus adatmodell leírást. Így ebben a fázisban már a multi-dimenzionális adatmodell szerinti leírást kell alkalmazni. Ekkor kerül meghatározásra, hogy milyen adatkockákra is van szükség. Az igényelt adatkockák körét a felmerült felhasználói igények alapján tudjuk meghatározni. Az egyes adatkockákhoz kapcsolódóan meg kell adni, hogy milyen információ elemeket foglal magába, s hogy ezek az információ elemek mely forrásokból és milyen lényegi átalakításokon keresztül hozhatók be az adattárházba. Ezen betöltési modul tervezési lépéshez kapcsolódik még a betöltéshez, adattisztításhoz kapcsolódó szoftver és hardver igények felmérése, az optimális konfiguráció megtervezése.

A durva tervezést követő részletes tervezési szakaszban kerülnek pontosításra az egyes adatkockák szerkezete. Ennek során meg kell adni, hogy milyen adattípusokat tartalmaz a rendszer, milyen dimenziókra van szükség, s milyen szerkezetűek lesznek az egyes változók és dimenziók. A részletes tervezés fázisának eredményeként előáll az implementációhoz szükséges rendszerterv, s ez alapján meg kell határozni, hogy milyen környezetben fog majd az implementálás lezajlani, ki kell alakítani az implementálási részre vonatkozó irányelveket, szabályokat. A rendszertervnek tartalmaznia kell az adatszerkezet leírása mellett az adatrendszerre vonatkozó integritási szabályokat, az elvégzendő műveletek, lekérdezések és adatfrissítések, körét, illetve magába foglalja az egyes alkalmazások adat lekérdezéseit, a programok működési környezetének megadását is.

| szint | DW elemek | műveletek | infrastruktúra |
|---|---|---|---|
| követelmény elemzés, megvalósítható ság | igényelt információ elemek | adatok forrásból való átemelése konverzió és integráció | HW és SW konfiguráció durva meghatározása |
| durva tervezés | multi dimenzionális adatmodell | átemelési rutinok, adatbetöltés, adat tisztítás | adatátemelés eszközigénye, közbenső adattárolás |
| részletes terv | adatkocka elemek pontos leírása | rutinok megvalósítási terve a fejlesztési környezet figyelembe vételével | implemenbtációs környezet kidolgozása |
| implementáció | adatbázis terv, | procedurák | konfiguráció |

| | | | |
|---|---|---|---|
| | indexek, dokumentációk | kódolása | megvalósítása, |
| tesztelés | DW rendszer ellenőrzési terve | tesztelési módszerek kidolgozása | tesztelési segédprogramok, szabványok |

Az implementációs részben kerül sor a programrendszer működéséhez szükséges hardver konfiguráció beszerzésére és telepítésérére. A beszerzéseknél az ár paraméter mellett figyelni kell az igényelt teljesítmény és szolgáltatási körre, a kapcsolódó szerviz szolgáltatásokra, a meglévő rendszerekkel való kompatibilitásra, a későbbi bővíthetőségre és a rendszer menet közbeni menedzselési igényeire. A hardver mellett a szoftver oldal kiépítési is ebben a lépcsőben történik meg. A szoftverek esetében a DW adatkezelő rendszert kiszolgáló adatbetöltő és adattisztító rutinok rendszerint nem állnak készen rendelkezésre, hanem a fejlesztési folyamat részeként ki kell dolgozni őket. A betöltési szakaszhoz hasonlóan a DW adatokat felhasználó lekérdező, döntéstámogatási feladatokat ellátó felhasználói programokat is itt kell kódolni. Az alkalmazások fejlesztése során valamely 4GL fejlesztő eszköz segítségével lehet a kódolási munkát hatékonyabbá tenni, gyorsítani.

A fejlesztés utolsó fázisában az elkészült rendszer helyességének az ellenőrzését végezzük. A programrendszer helyességének ellenőrzése rendszerint igen bonyolult feladat, hiszen a programrendszer több egymástól függő modulból áll, s az egyes modulok több elágazást, ciklus elemet is tartalmaznak, így nem lehet minden lehetséges program lefutási ágat egyenként ellenőrizni. Ezért ezen fázis részeként ki kell választani, hogy mely tesztelési technika lesz a legmegfelelőbb az implementált rendszer vonatkozásában. A tesztelés során célszerű mind a verifikáció mind a validáció lépéseire kitérni. A verifikáció során az implementáció nyelvtani, formális helyességét ellenőrizzük, míg a validációnál a tartalmi helyesség ellenőrzése a cél. A két vetületet összevetve a verifikáció lépéséhez áll rendelkezésre több segédeszköz, mivel az problémakör függetlenül is értelmezhető, vizsgálható.

A felsorolt lépéseket együttesen magába foglaló fejlesztési mátrixot szem előtt tartva pontosabban megbecsülhető a fejlesztés várható költség és időráfordítása is. A DW rendszer sikeres megvalósítása a megfelelő előkészítés, menedzsment mellett az implementáció színvonalától is jelentősen függ. Ezért a DW részletes tervezése során törekedni kell az adatmodell és a fejlesztő rendszer által biztosított lehetőségek minél jobb kihasználására. A következő részben ezért újból visszatérünk a DW adatkocka adatmodelljének tárgyalására, de most már a speciálisabb modellezési lehetőségeket vesszük sorra, mely segítségével hatékonyabbá, robosztusabbá tehető az elkészült adatrendszer.

## Multi-dimenzionális adatmodell speciális elemi

Egy adattárház rendszer az érintett szervezet teljes vertikumát átfogó információs rendszerként kezelendő, melyben több szervezeti egység adatai is helyet foglalnak. A DW azonban nemcsak befogadja a különböző helyről származó adatokat, hanem integrálja is őket, egy egységes adatrendszert alkotva. Ez az egységesség azonban sokszor nem magától értetődően jön, hanem tudatos tervezés eredménye. Ha veszünk például egy nagy nemzetközi céget, melynek több országban is lehet telephelye, akkor a cég DW rendszere több országra vonatkozó adatokat is tartalmazni fog. A DW rendszerek rendszerint a helyi DataMart

rendszerekből fejlődnek ki. Ennek megfelelően a DW adatkockáinak tervezésekor alapul lehet venni a meglévő DataMart rendszerek adatkocka modelljeit. A példaként vett vállalat esetében az egyes részlegek igényeit kielégítő adatkockák fogják alkotni a teljes vállalati DW rendszert. Feltehetjük, hogy ezen részleg szintű modellekben, melyek lokálisan kerültek kidolgozásra, mindenhol találkozhatunk a vevők, a partnerek adatainak letárolása kapcsán az elérési cím adatokkal. E címek dimenzió szerepkörben fognak megjelenni az egyes adatkockáknál, utalva a város, régió pozícióra is. A címet megadó dimenziót nevezzük el ELERESICIM-nek.

Az ELERESICIM dimenzió tehát minden egyes integrálandó DataMart részben szerepel, így áthozhatók a közös integrált rendszerbe is minden részleg esetében. Azonban ez a beintegrálási folyamat nem is olyan egyszerű, mint az első pillantásra tűnik. Ha ugyanis pontosabban megnézzük, az ELERESICIM dimenzió nem szükségszerűen jelenti pontosan ugyanazt a fogalmat minden egyes részlegnél. Előfordulhat ugyanis, hogy az egyik részlegben a cím a

- megye
- telepules
- utca
- hazszam

adatokat foglalja magába, míg egy másik helyen ugyanezen dimenzió név alatt az

- ország
- irányítószám
- település
- hazszam
- telefon

mezők adatai kerülnek letárolásra. Ha ellenőrzés nélkül átvesszük mindkét dimenziót, s meghagyjuk eredeti értelmezésüket, akkor a rendszerünkben az integráció után az ELERESICIM dimenzió több adatkockában is szerepelni fog, de a különböző adatkockákban más és más jelentéssel. Ez azonban már igen veszélyes dolog, hiszen egy külső szemlélő számára, aki az integrált rendszerrel találkozik, az azonos elnevezés azonos tartalmat sugall, teljesen jogosan. Az azonos tartalomra és struktúrára építve így a két kocka integrálása is természetes lépésnek tűnik, azonban az eltérő szerkezet miatt ez a lépés sem valósítható meg. Az azonos elnevezés és eltérő jelentés sok esetben okozhat komoly bonyodalmat. Ezért a tervezést úgy kell végig vinni, hogy kiküszöböljük az ilyen jellegű ellentmondásokat.

A fentiek alapján tehát olyan dimenziókat kell alkalmazni az adattárház rendszerben, amelyek minden részben, minden adatkockában ugyanolyan jelentéssel és szerkezettel rendelkeznek. Az ilyen egységes jelentéssel és szerkezettel rendelkező dimenziókat nevezik összeférő (conformed) dimenzióknak. Az összeférő dimenziók meghatározása tehát alapvető fontosságú a több részterületet lefedő adatrendszer kialakításakor, hiszen az összeférő dimenziók mintegy előfeltételnek is tekinthetők a különböző területekről származó adatok integrálásának. A tartalmi és formai illeszkedés vizsgálatát és kialakítását minden dimenzió esetében el kell végezni, hiszen a legjobban magától értetődőnek tűnő dimenziók, fogalmak értelmezése is eltérhet a különböző részrendszerek alkalmazói csoportjai között.

A dimenzió mellett a többi adatmodell elemre is értelmezhető az illeszkedés, a konformitás fogalma. Így a változók esetében is beszélhetünk összeférő változókról, amikor is az azonos változó elnevezés mögött azonos jelentés és azonos szerkezet húzódik meg az adattárház modell minden adatkockája esetén. Az összeférő változók és dimenziók alkalmazása nemcsak logikai lezártságot jelent, hanem egyben szükséges előfeltétele is a különböző data mart vagy

adatbázis forrásokból származó adatelemek integrálásának. Az egységes jelentés és forma kidolgozása természetesen többlet energiát és munkát jelent a fejlesztés oldaláról nézve, s sokszor nem is olyan egyszerű és gyors feladat, mint azt első pillantásra gondolnánk. Itt elsősorban a figyelembe vehető esetek nagy száma, az egyes esetek felderítése okoz problémát. Igen tanulságos példát mutat be erre a feladatra a Kimball () könyv, melyben az ügyfelek név és cím adatainak leírására szolgáló dimenzió mezőket kell meghatározni.

A név és cím adatok tárolása esetén a jó rendszermodellnek fel kell készülnie minden lehetséges alkalmazási feladatra, melyhez ezen adatokra szükség lehet. Ide tartozik többek között a

- megszólítás
- levél címzés
- telefonálás
- fax küldés
- e-mail küldés
- intézmény azonosítás
- végzettség, beosztás szerinti osztályozások

Az információ igényeket összegyűjtésénél arra is gondolni kell, hogy az ügyfelek nemcsak egy országból, régióból jöhetnek, ezért fel kell deríteni azt is, hogy az egyes cím elemek milyen eltérő tartalommal és formátummal rendelkezhetnek a különböző régiókban. Minden lehetséges esetet figyelembe véve igen terebélyes lesz azon információ és adatelemek száma, melyekre szükség lehet a megszólítás és cím adatoknál.

Az adatszerkezet kialakításakor a legegyszerűbb megoldásnak az tűnhet, hogy a dimenzió mögött csak néhány mezőt veszünk fel, melyek tartalma a körülményeknek megfelelő formátumban tölthető fel. Egy ilyen egyszerűsített szerkezet lehet például az alábbi struktúra:

- név
- beosztás
- vállalat
- ország
- lakcím
- telefon
- fax
- e-mail

A fenti struktúra ugyan egyszerűen áttekinthető, azonban a feldolgozás során a struktúra egyszerűsége a hatékonyság és a teljesítőképesség jelentős romlását vonhatja maga után. Ha ugyanis levél megszólítást kell készíteni, akkor a fenti adatok már nem megfelelőek, nem lehet automatikusan feltárni, hogy férfiről vagy nőről van-e szó, s milyen egyéb titulust illik tenni a név elé. A fenti probléma megoldásában nem sokat könnyítene az a változat sem, amikor a név magába foglalja a titulust is. Hiszen gondoljunk bele, mennyi különböző előtag fordulhat elő a különböző országokban, s egy országon belül is hány különböző módja lehet a kiegészítő tagok megadásának. A sokszínűség illusztrálására álljon itt egy kis ízelítő a lehetséges variánsok halmazából:

- Mrs R. Jane Smith,

- Frau Helena Smidt
- Kelemen Zoltánné
- Kelemenné Nagy Mária
- Kelemen Zoltán Béla
- Kelemen Zoltán, dr
- dr Kelemen Zoltán
- ifj. Kelemen Zoltán dr
- dr ifj Kelemen Zoltán

Elképzelhető, hogy milyen időigényes lehet olyan program, algoritmus készítése, amely az összes lehetséges esetre felkészülve ki tudja venni a cím szövegből a családi név, a keresztnév, a nem, az előtag részeket. Ehhez hasonló problémát jelenteni az az eset is, amikor osztályozni szeretnénk az ügyfeleket az egyes városok szerint. Ugyan város megnevezés benne van a lakcím részben, de a lakcím a teljes lakcímet tartalmazza több különböző komponensével. A város meghatározásához tehát itt is el kell tudni különíteni a városra utaló szót a cím többi részétől. Ez az előző problémával összemérhető nehézségű feladat. Természetesen, ha a város részre sehol sem szükség a későbbiekben sem, akkor felesleges további bontást eszközölni, s maradhat az eredetileg tervezett több elemet is magába foglaló cím mező a dimenzió sémában.

A feldolgozási nehézségekre való tekintettel célszerű tehát az információkat olyan elemi szintű adategységekre bontani, amelyeket a későbbi feldolgozási algoritmusok igényelnek. Az elemekre való bontás ugyan meg fogja növelni a séma méretét, s elsőre összetettebbnek tűnik maja a modell, azonban ezáltal sokkal egyszerűbb és hatékonyabb feldolgozó algoritmusok hozhatók létre.

Az előzőekben említett példához visszatérve, a felbontások eredményeképpen a javasolt dimenzió szerkezet a () alapján a következő:

- megszólítás
- üdvözlési forma
- informális megszólítás
- családi név
- keresztnév
- előtagok
- név nemzetisége
- nem
- rang
- pozíció
- szervezet
- alszervezet
- másodlagos al-szervezet
- házszám
- utca
- utca típusa
- kerület
- postafiók
- másodlagos kerület jelző
- megye
- ország

- földrész
- irányítószám
- másodlagos irányítószám
- irányítószám típusa
- telefon országkód
- telefon terület kód
- telefonszám
- mellék
- fax országkód
- fax terület kód
- faxszám
- e-mail
- web-lap
- nyilvános kulcs

Látható tehát, hogy a valóban igen alaposan kell elemezni minden egyes dimenziót, változót az integrált adattárház rendszer kialakításakor, hogy minden szükséges információ a legjobban kezelhető formában rendelkezésre álljon. A dimenzió és változó struktúrájának kialakításakor így több szempontot is szem előtt kell tartanunk a tervezés során, melyek figyelembe vétele biztosítja, hogy a megfelelő struktúra fog előállni. Ezen szempontok közül a legfontosabbakat az alábbiakban foglaljuk össze:

- leíró elnevezés: az elnevezésnek sugallnia kell a tartalmat, hogy a felhasználók a séma áttekintése után el tudják dönteni, hogy mennyire hasznosak az egyes dimenziók vagy változók a vizsgált probléma esetében.
- általános jelentésű legyen: a struktúrának minden lehetséges alkalmazási feladat és körülmény között alkalmazhatónak kell lennie. Nem szabad csak egy speciális esetre gondolva megszabni a struktúra elemeket, mint azt egy előző példában is bemutattuk, mivel ebben az esetben jelentősen leszűkítjük a kapcsolódó alkalmazási területek körét.
- tiszta, helyes ellenőrzött értékű legyen. A változók és dimenziók közé csak olyan mennyiségeket szabad felvenni, melyek tartalma egyértelmű, ellenőrzött, s a vizsgált feladok szempontjából relevánsak.
- jól tagolt: mivel az alkalmazások, a feldolgozások szempontjából az az előnyös, ha az igényelt adatelemek már készen rendelkezésre állnak, s nincs szükség bonyolult elemzési, kiemelési funkciókra. Ehhez azonban a tárolt információt elemi információkra és elemi adatelemekre célszerű feldarabolni.
- hatékonyan kezelhető, indexelhető: a belső karbantartás hatékonysága szempontjából célszerű, ha az azonosító szerepet betöltő elemek kis méretűek és egyértelmű rendezés értelmezhető rajtuk.

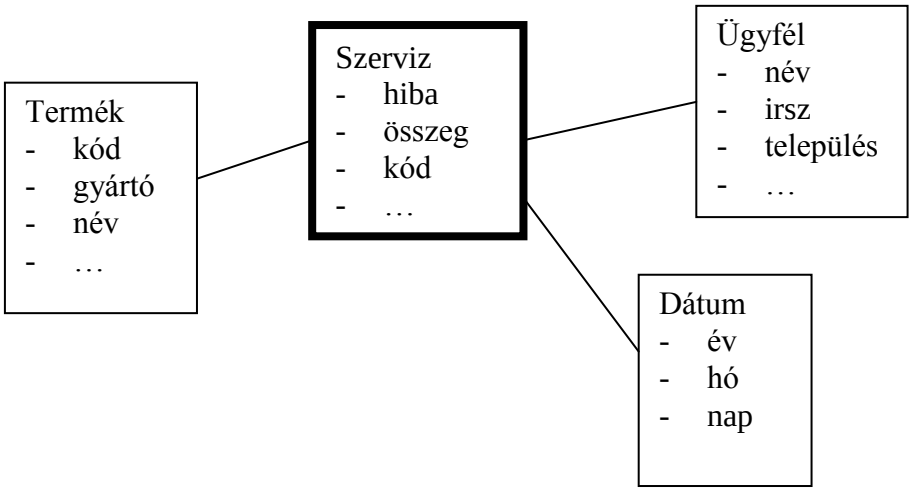
A dimenziók tervezése során a fenti irányelvek betartása mellett is előfordulhatnak olyan esetek, amikor a szokásostól eltérő struktúrákat kell létrehozni. Egy ilyen speciális dimenzió típus a degenerált dimenzió fogalma.

A degenerált dimenzió olyan dimenziót jelent, amelynek nincs attribútuma. A dimenziókat ugyanis eddig úgy vettük, hogy van egy azonosító elnevezése, s egy leíró struktúrája, amely az attribútumait adja meg. Egy ilyen dimenzió lehet például az ügyfél:

ÜGYFEL

- név
- irányítószám
- település
- kerület
- utca
- utca jelleg
- házszám
- lépcsőház
- emelet
- ajtó

A tulajdonságok adják meg a dimenzió előfordulás leírását, adatait. Az ügyféldimenzió például tagja lehet egy TV szerviz szolgáltatásokat nyilvántartó adattárház rendszernek. A lehetséges kapcsolódó adatkockák közül az SZERVIZ-et kiemelve az ide tartozó adatkocka modell az alábbi alakot ölti.



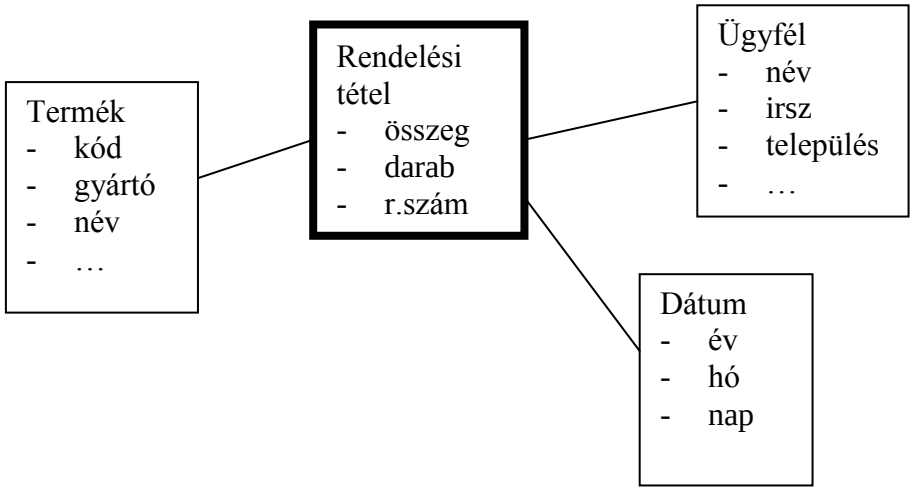

26. ábra, Szerviz rendszer szemantikai modellje

Ha most példaként egy vállalati rendelés nyilvántartó adattárházat veszünk, amelyben a rendelés központi szerepet foglal el, s a rendelésről a

- rendelési tételek
- ügyfél
- dátum

adatokat tartjuk nyilván, akkor az alábbi modell rajzolható meg a sémához:

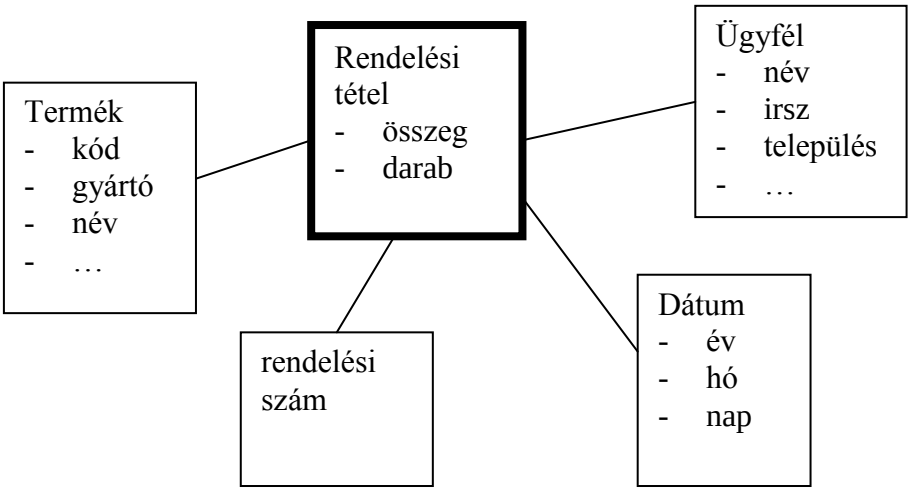
A kapcsolódó adatkocka séma egyik jellegzetessége, hogy a rendelési szám egy speciális szerepben jelenik meg. Egyrészt szükség van a rendelési számra, mivel fontos információt hordoz, s a felhasználók igénylik a meglétét. Mivel a rendelési szám nem egy additiv mennyiség, hanem egy azonosító jelleggel bíró mennyiség, ezért nem célszerű a rendelési számot változó, az adatcella tulajdonságaként szerepeltetni. Ha viszont dimenzióként vesszük fel, akkor neki nem fogunk találni értelmes struktúrát, hiszen a tényleges jellemzők mindegyike kikerülhet önálló dimenzióba, így dimenzió lesz a




27. ábra, Rendelési rendszer szemantikai modellje

- ügyfél
- dátum
- termék

Ennek megfelelően az alábbi MD séma készíthető el:

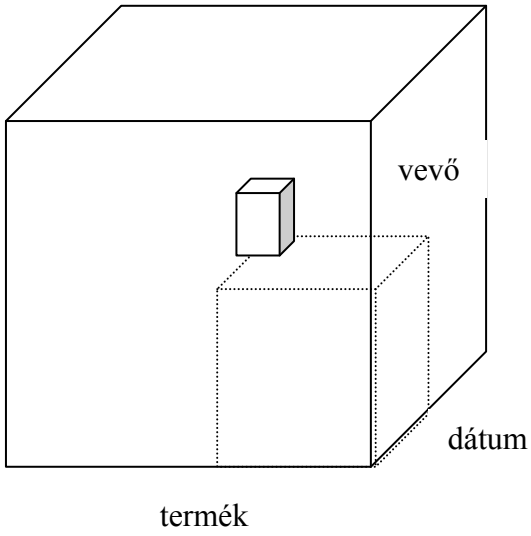



28. ábra, Rendelési rendszer degenerált dimenzióval

Az így létrejött sémában a rendelési szám egy tulajdonság nélküli dimenzió. Az ilyen, tulajdonság nélküli dimenziókat szokás degenerált dimenziónak nevezni. A degenerált dimenzió jele a szemantikai modell szintjén az üres téglalap. A degenerált dimenziók fizikai megvalósítását illetően, ha a szokásos struktúrát alkalmazzuk, akkor a ténytáblában a cellát leíró adatok között szerepelni kell egy kapcsoló szerepet betöltő mezőnek, amely kijelöli, hogy az adott cella mely rendelési szám dimenzió előforduláshoz kapcsolódik. Ez a kapcsoló mező rámutat, kijelöl egy dimenzió előfordulást leíró rekordot. A rendelési szám esetében ez a leíró rész csak magát az azonosító szerepet betöltő rendelési szám értéket tartalmazza, azon kívül semmi más elem nincs benne. Mivel a kapcsoló, hivatkozó mező a tény cellában szintén csak ezen rendelési szám értéket tartalmazhatja, ezért a különálló dimenzió előfordulást leíró

rész nem hordoz semmilyen újabb információt, redundáns, felesleges adatelem. Emiatt nem szokás a degenerált dimenziók fizikai megvalósításánál külön dimenzió előfordulás rekordokat is felvenni, fizikai szinten a degenerált dimenziók csak a ténytáblában fordulnak elő.

A szokásos dimenzió fogalom esetében a dimenzió a tény változó egyed egy jellemzőjét tartalmazza, kijelölve minden tény cellához egy dimenzió előfordulást. Ennek megfelelően az egyes változó előfordulások egyértelműen meghatározhatók a hozzá tartozó dimenzió előfordulásokkal. A dimenziók egyértelműen kijelölik az egyes ténycellák pozícióját az adatkockán belül, mint azt a következő ábra is mutatja egy rendelési tétel adatkocka esetében, ahol minden rendelési tételhez egyértelműen tartozik egy vevő, egy dátum és egy termék dimenzió érték.

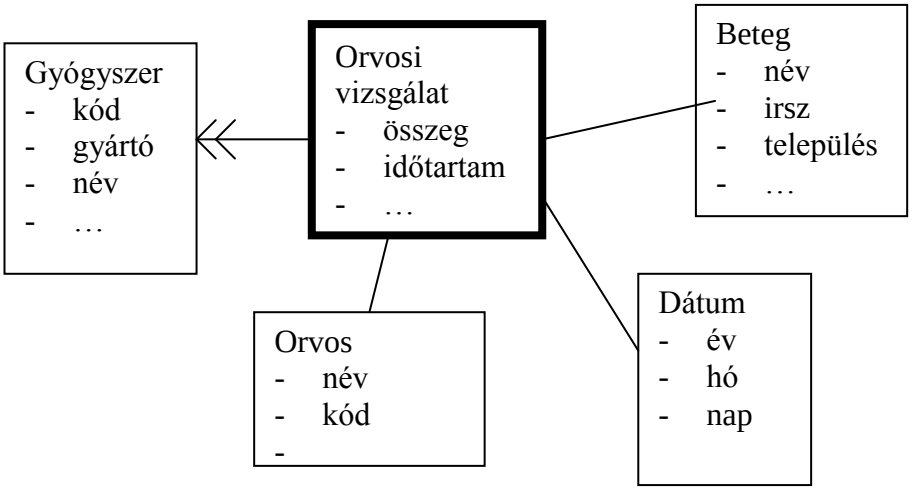



29. ábra, Adatkocka dimenzióival

Ha azonban a rendelési tétel adatkocka helyett egy rendelés adatkockát, vagy egy orvosi vizsgálat leíró adatkockát veszünk, akkor ott az előbb felvázolt szerkezet nem érvényes, hiszen itt már nem igaz az a feltevés, hogy minden tény cellához csak egyetlen egy dimenzió előfordulás köthető. A rendelés esetében például egy rendeléshez több termék is köthető, illetve egy orvosi vizsgálathoz több feltárt betegség is köthető. Ezen esetekben a dimenzióknak több előfordulása is társítható az egyes ténycellákhoz. Az ilyen jellegű dimenziókat szokás többértékű dimenzióknak nevezni.

A szemantikai modell szintjén a többértékű dimenziókat a kapcsolatot jelölő vonal dupla nyilazásával jelölhetjük ki, mint azt a következő ábra is mutatja.

Ha a fizikai szinthez közeli adatmodellre térünk rá, akkor itt is látható, hogy a szokásos megvalósítás, melyben minden ténycellában egy kapcsoló szerepet betöltő mező foglal helyet a megfelelő dimenzió előfordulásra mutatva, nem járható. Ugyanis most a ténycellát leíró struktúrában több dimenzió előfordulásra kellene mutatni, amihez több kapcsolat leíró mezőre lenne szükség. Az az út, hogy megismételjük a kapcsolat leíró mezőket annyiszor, ahány kapcsolódó dimenzió érték van, nem ajánlott, ugyanis ekkor a sémában egy előre megadott darabszámú kapcsoló mezőnek kellene helyet foglalni, s ezen mezők számát nem lehet pontosan megbecsülni előre, hiszen itt egy cellánként változó számú kapcsolatról van szó.




30. ábra, Adatkocka többértékű dimenzióval

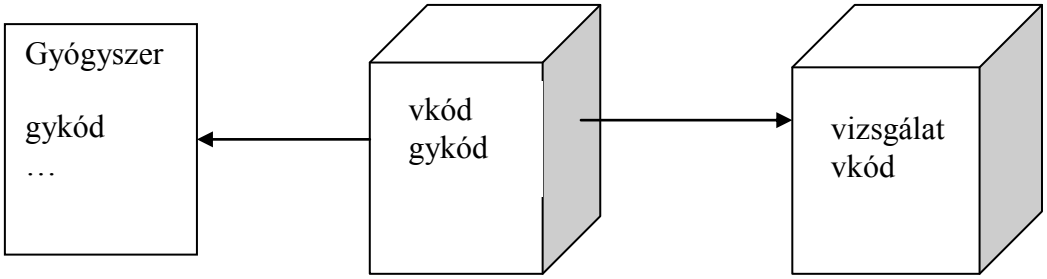
Az itt felvázolt probléma nemcsak a multi-dimenzionális modellben fordul elő, hanem a hagyományos relációs adatbázis modellben is, hiszen ott is csak egyértékű lehet a kapcsoló mező. Az ilyenkor szokásos megoldás a relációs modellben egy külön kapcsoló tábla bevezetése, s ez az út itt is járhatónak bizonyul. Ehhez fel kell venni egy külön táblát, amelynek célja a kapcsolódó cella előfordulás és dimenzió előfordulás párosok nyilvántartása. Az ilyen kapcsoló szerepet betöltő táblákat az MD modellezésben híd (bridge) tábláknak nevezik.

A híd tábláknak kell kijelölnie a kapcsolódó párosokat, ezért a szerkezetüknek olyannak kell lennie, hogy abból a kapcsolódó páros azonossága kiderüljön. Így a tábla cellájában szerepelnie kell egy dimenzió előfordulást azonosító mezőnek és egy ténycella előfordulást azonosító résznek. Mivel a dimenzióknak létezik ilyen azonosító szerepű mezője, hiszen ezt használta volna eredetileg a ténycella is, ezért az ilyen irányú kijelöléssel nincs probléma. A másik irány esetén azonban azt láthatjuk, hogy alapesetben a ténycellának nincs szüksége külön egyedi azonosító elemre, hiszen azt a hozzá kapcsolódó dimenzió előfordulások fognak azonosítani. Ezért itt egy külön mező bevezetésére van szükség, amely alkalmas a kapcsolódó cella kijelölésére. Ezen feladat egyik szokásos megoldása, amikor egy új, csoportkulcsnak nevezett mezőt illesztünk be a ténycellát leíró tulajdonságok közé. Minden egyes ténycellának lesz egy egyedi csoportkulcsa, s a híd tábla cellái az összetartozó dimenzió kulcsot és csoportkulcsot tartalmazzák.

A fenti két kapcsoló szerepet betöltő mező mellet a híd cella helyet adhat más, a kapcsolódó párost jellemző tulajdonságnak is. Ha például a rendelés adatmodellben nyílván kívánjuk tartani többek között az alábbi adatokat:

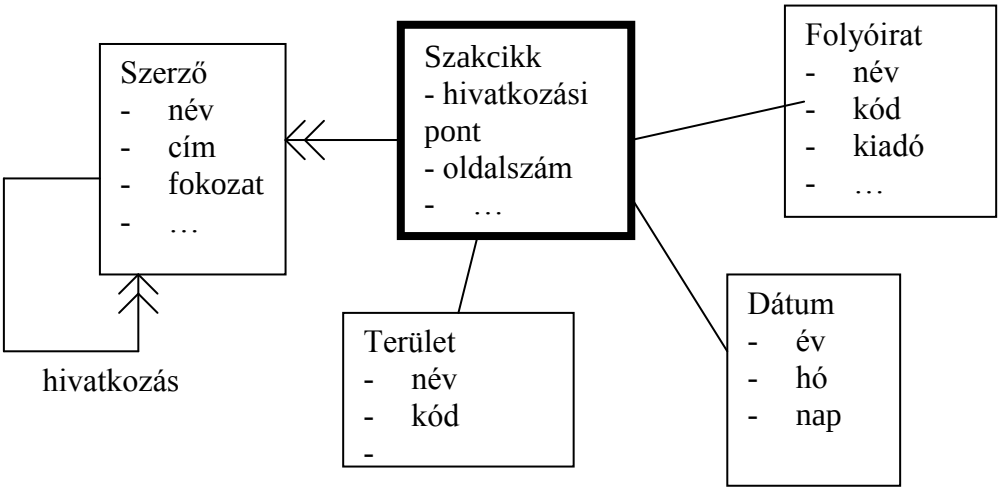
- rendelés összértéke
- rendelet áru mennyisége tételenként

akkor a két felsorolt tulajdonság közül az első a rendelésre vonatkozik, míg a másik a rendelési tételre, amelyből egy rendeléshez több is tartozhat. Ezért a rendelt termék darabszáma a termék-rendelés pároshoz tartozik, így elhelyezése a párost leíró híd táblában lehetséges a meglévő kapcsoló mezők mellé. A rendelési példánál maradva az eredményül kapott, s a híd táblát is tartalmazó adatmodell leírást a következő ábra mutatja be.




31. ábra, Híd tábla alkalmazása

A híd táblát egyébként nemcsak a többértékű dimenziók esetében szokás használni, hanem a hálós dimenziók esetében is, hiszen itt sem közvetlenül mutat a rendszer a tény cellából a hozzá tartozó dimenzió előfordulás hierarchia miden tagjára. Példaként tekintsünk egy kutatási cikk gyűjtemény adatbázist, melyben az egyes szerzők is a nyilvántartásban szerepelnek a közöttük fennálló hivatkozási kapcsolattal együtt.
A szemantikai adatmodellben ez a felbontás az alábbi módon jelenik meg:

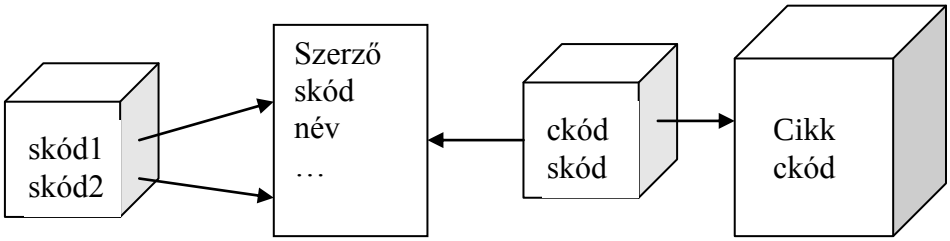



32. ábra, Szakcikk adatkocka modellje

Mivel itt sem ismert előre, hogy milyen mélységű is lesz a dimenzió háló, ezért most sem lehet fixen beépíteni a cella struktúrába az összes dimenzió előfordulásra mutató pointert. A tetszőleges méretű dimenzió háló kezelésére ezért jelen esetben is a híd kocka alkalmazása a legkézenfekvőbb megoldás, mivel segítségével rugalmasan leírhatók a változó méretű kapcsolat rendszerek is.

Az egyes híd tábla cella előfordulások az egyes dimenzió előfordulások kapcsolatát adják meg, minden egyes leszármazotti viszonyhoz egy híd táblabeli cella előfordulás fog létrejönni. A híd cella előfordulás két kapcsoló mezőt tartalmaz, az egyik mutat az ős dimenzió előfordulásra, a másik pedig annak egy leszármazottjára. Technikai okokból célszerű olyan híd cellát is létrehozni, melyben mindkét kapcsoló ugyanazon dimenzió előfordulásra mutat. Ekkor az induló ténytábla cella egy ilyen híd cella előfordulásra fog mutatni, mely minkét kapcsoló mezőjével az érintett dimenzió előfordulást fogja tartalmazni. A rendszer tartalmazni fog még több olyan híd cella előfordulást is, melyek ezen induló

dimenzió előfordulást a belőle származó dimenzió előfordulásokkal kötik össze. Az így megalkotott struktúrában az előfordulások kapcsolódását a következő ábra mutatja be.




33. ábra, Híd adatkockák alkalmazása

A dimenzió kezelés egy más jellegű speciális esete az amikor egy dimenzió típus többszörözötten is kapcsolódhat egy ténytáblához. Ha veszünk például egy szervízelést nyilvántartó adatkockát, akkor az adatcellában leírt szervízelési folyamathoz többféle szerepkörben is hozzárendelhetjük az idő dimenziót, nevezetesen, mint

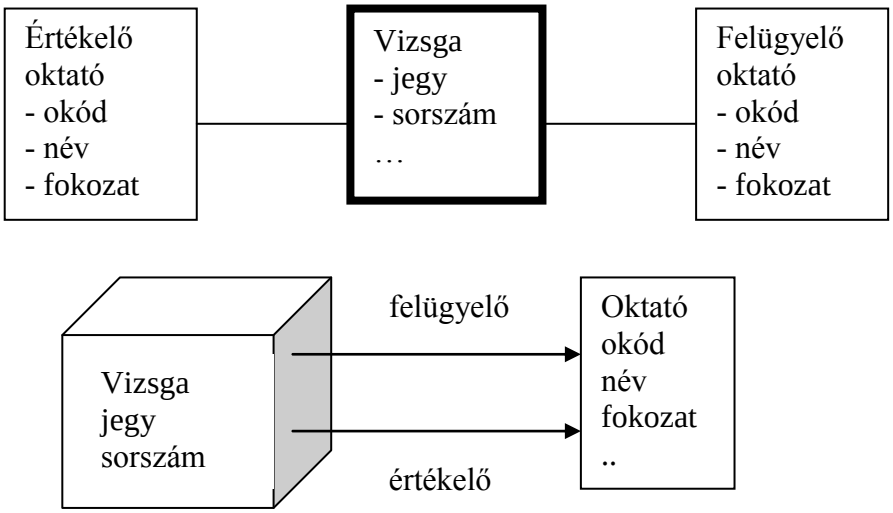
- hiba bejelentés időpontja
- kiszállás időpontja
- eszköz beszállítás időpontja
- munka elkezdés időpontja
- munka befejezés időpontja
- termék átadás időpontja
- számla kifizetés időpontja

A fenti elemek mindegyikét lehet egy-egy önálló dimenzióként szerepeltetni az adatmodellben, hiszen mindegyik más-más jelentéssel bír. A szokásos megvalósítás szerint ekkor mindegyik dimenzió mögött egy önálló dimenzió előfordulás táblázat állna. A jelen esetünkben ez azt jelentené, hogy az időpontok megadása több helyen is ismétlődne. Vagyis fizikailag egy megadott időpont leírása több helyen is előfordulna redundanciát okozva. A rendszer jobb helykihasználása végett azonban hasznosabb megoldásnak bizonyul, ha magát az idő adatokat csak egyszer tároljuk le, s a különböző dimenziók esetében közösen kerülnek felhasználásra az egyes dimenzió előfordulások. Ekkor egy fizikai dimenzió több különböző logikai szerepkörben is megjelenik az adatkocka modellben. Ezt a jelenséget a szerepkörökkel ellátott dimenziók esetének nevezzük.

A szerepkörökkel ellátott dimenziók esetében egy fizikai dimenzió több különböző szerepkörben is megjelenhet az egyes tény táblákban, viszont mindegyik szerepkör esetén a kapcsoló hivatkozások egy közös dimenzió előfordulás halmazba mutatnak, a dimenzió előfordulásai csak egyszer, redundancia nélkül kerülnek letárolásra. Ezáltal jelentős helytakarékosság érhető el, másrészt sokkal hatékonyabb lesz az egyes szerepkörökön keresztüli összekapcsolások megvalósítása, hiszen nem kell különböző táblák rekordjait illeszteni, hiszen egy táblában helyet foglal az összes előfordulás. A szerepkörökkel ellátott dimenziók megvalósítását szemlélteti a következő ábra.

A hagyományos adatbázis kezelés keretében megszoktuk, hogy az adatbázisunk a vizsgált rendszer aktuális állapotát mutatja, az egyes adatbázismezőknek, tulajdonságoknak egyértelmű, egyedi értékük van, amely az vizsgált rendszerbeli aktuális értéket mutatja. Ha

például van egy bolti nyilvántartásunk, melyben a forgalom nyilvántartása is szerepel, a termék leírásban szereplő ár érték az aktuális ár értéket jelenti, mely alapján tudja a rendszer kiállítani a számlát.




34. ábra, Szerepkörrel ellátott dimenziók

Ha változik az áru ára, akkor az adatbázisban a megfelelő mező értéke módosul, a régi érték elvész, s csak az új érték marad meg. Emiatt a korábbi állapotok már nem következtethetők vissza az adatbázisban tárolt értékekből.

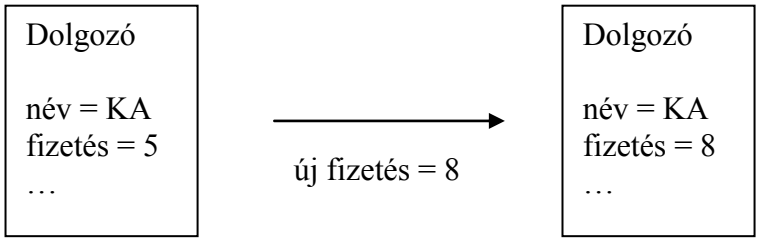
Ez a fajta adattárolási mechanizmus igen helytakarékos, s a legtöbb információs rendszerben meg is felel a célnak, hiszen csak az aktuális állapotokra van szükség. Az adattárházak esetében azonban egy másfajta megközelítésre van szükség, hiszen az adattárházak egyik alapjellemzője, hogy az adatokat történetiségükben ábrázolják. Emiatt nyilván kell tartani az aktuális értékek mellett a korábbi értékeket is.

A dimenzió értékek módosulásának kezelésére a DW rendszerek többféle megoldást is kínálhatnak, attól függően, hogy mennyire pontosan tartják be a történetiségre vonatkozó követelményeket. Tipikusan három alapmegoldás terjedt el:

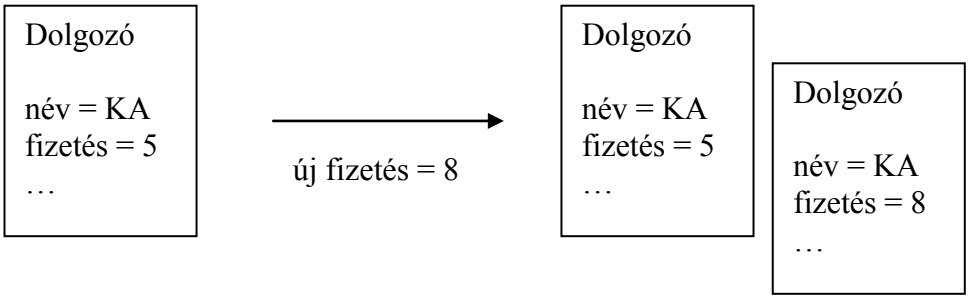
- a rendszer nem törődik a régi értékek, a történetiség megőrzésével, a dimenzió érték változásakor a korábbi érték felülíródik a hagyományos OLTP rendszerekhez hasonlóan.
- A DW alapkoncepciójának megfelelően a rendszer a minden változást nyilvántart, az egyes verziók mind megmaradnak a dimenzióhoz kötötten, megadva hogy mely időszakra voltak érvényesek. Egy dimenzió értékhez több verzió rekord, egy dimenzió history kötődik.
- A harmadik változatban egyetlen dimenzió rekord van jelen, azonban a rekordon belül több mező is szerepel egy tulajdonsághoz kötötten. Ekkor egy rekordon belül együttesen kerülnek nyilvántartásra a régi és az új adatok.

Az egyes változatok eltérő helyigénnyel és eltérő funkcionalitással rendelkeznek. Az első megoldás esetén csak az aktuális érték tárolására kell helyet biztosítani, viszont itt a történetiség kezelése teljes egészében kimarad. A második változatban legnagyobb a

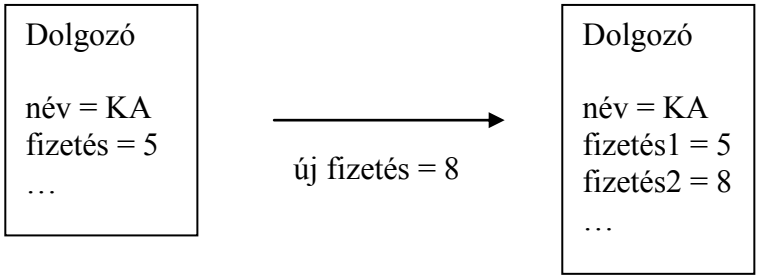
helyigény, hiszem egy adatelemnek több history adata is letárolódik. Ezáltal azonban lehetőség nyílik arra, hogy a múltbeli eseményeket, állapotokat is vissza lehessen idézni. A közbenső megoldás esetén a változó méretű dimenzió rekordok problémáját kell megoldani. A három változat összehasonlítását mutatja be a következő ábra.




35. ábra, Dimenzió módosulás helyettesítéssel




36. ábra, Dimenzió módosulás rekord verzió őrzéssel



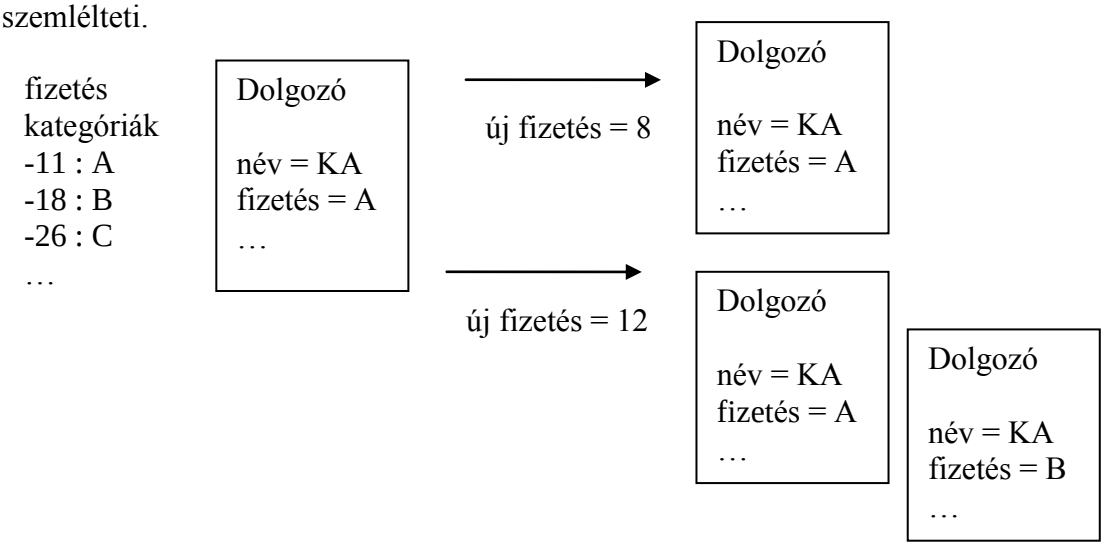

37. ábra, Dimenzió módosulás tulajdonság verzió őrzéssel

Ugyan a funkcionalitást tekintve a teljes történetiség nyilvántartása lenne a legjobb megoldás, azonban a felmerülő extra helyigény miatt szokták a másik két módszer valamelyikét is. A pótlólagos helyigény annál nagyobb minél gyakoribb a dimenzió érték módosulása. Ezért a gyakran változó értékek esetén kiegészítő mechanizmusok terjedtek el a méretek elfogadható keretek között tartására. Az egyik megoldás szeparálja a kis számú gyakran változó tulajdonságot a többi, ritkán változó tulajdonságtól. A másik megoldás az érték változások gyakoriságát próbálja meg csökkenteni egy intervallum-érték tárolásra történő áttéréssel.

A tulajdonság szeparáció azon a megfontoláson alapszik, hogy a szétválasztás után a két tulajdonság rekord részből a változások csak a kisebb terjedelmű részt fogják érinteni, így csak azt a részt kell megismételni a history megőrzése során. Ezáltal a különböző dimenzió változatok letárolása kevesebb helyet fog igényelni, mintha a teljes dimenzió rekordot

letároltuk volna. E takarékossági megoldás természetesen csak akkor jár jelentős helymegtakarással, ha sikerül a dimenzió tulajdonságokból kiválasztani egy olyan kisebb terjedelmű csoportot, melyek jelentősen gyakrabban módosulnak, mint a többi tulajdonság.

A másik takarékossági módszer alapelve, hogy a gyakran változó tulajdonságok esetében az egyedi értékek helyett egy intervallumértéket tárolnak le. Így például egy páciens nyilvántartás esetén, melyben páciens dimenziónál a testsúly is letárolásra kerül, a testsúly egy gyakran változó tulajdonság lehet, melynek pontos értéke rendszerint nem is annyira fontos, elegendő csak egy közelítő érték ismerete. Ebben az esetben a testsúly lehetséges értékeinek halmazát felosztjuk intervallumokra, s a dimenzió rekordban a tulajdonsághoz a pontos érték helyett az értéket befoglaló intervallumot adjuk meg. Mivel a testsúly apróbb változásai során a befoglaló intervallum nem fog változni, ezért a adattárházban sem kell újabb dimenzió változatot felvenni. Új változat létrehozására így csak akkor van szükség, ha az érték jelentősen változik. Ez a módszer tehát a változások gyakoriságának lecsökkentésével redukálja az igényelt tárhelyet. A következő ábra e módszer mechanizmusát szemlélteti.




38. ábra, Dimenzió módosulás intervallum képzésnél

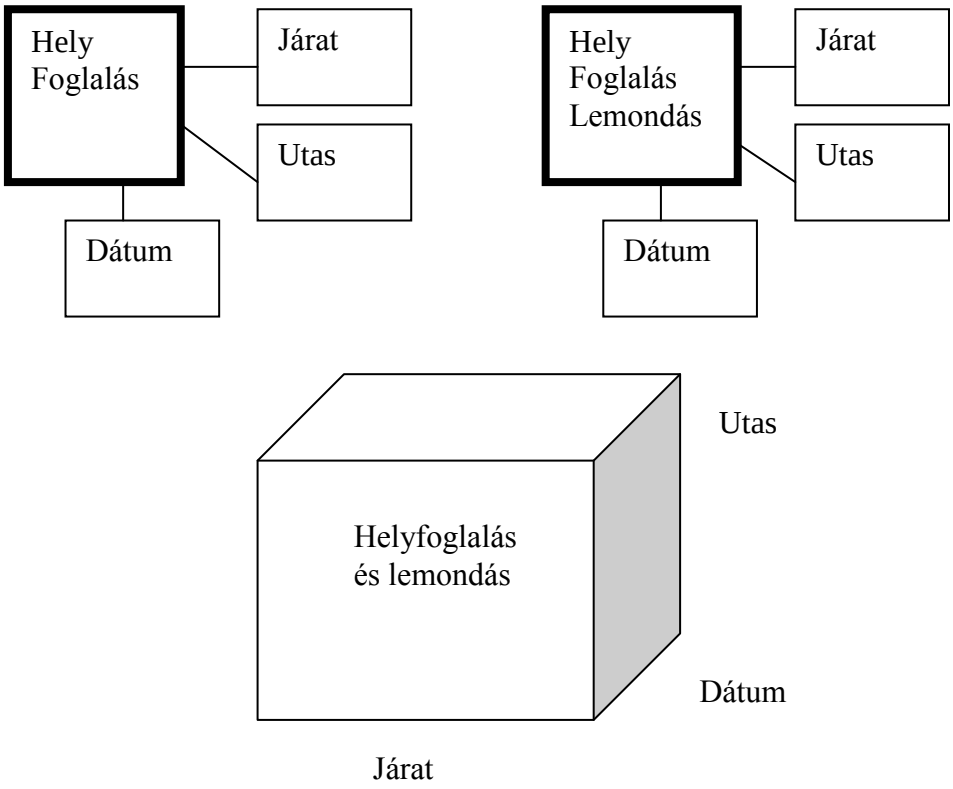
Az előző probléma elemzése során már látható volt, hogy az adattárház tervezése során a teljesség, helyesség követelménye mellett mennyire fontos a hatékonysági szempontok figyelembe vétele is. A hatékony megvalósítást elősegítő modellezés mellett maga a DW rendszer motorja is tartalmaz egy sor olyan mechanizmust, mellyel jelentősen lecsökkenthető az OLAP lekérdezések végrehajtási költségei. A következő fejezetben e mechanizmusokat fogjuk áttekinteni.

## Adatkocka műveletek hatékony végrehajtása

Mint már korábban említettük, több forrásból integrált adatrendszerek esetén az összeférő dimenziók használata igen lényeges modellezési technika. Ilyen összeférő dimenziók esetén több adatkocka is hasonló dimenzió struktúrát mutathat fel. A hasonló szerkezet jellegzetes példája, amikor több, egymással strukturális kapcsolatban álló változót tartanak nyilván az adattárházban. Strukturális kapcsolatként jelenik meg a modellben a specializáció – általánosítás viszonya is. Ha van például veszünk egy banki alkalmazást ahol a számlákhoz több különböző tranzakció típus is tartozhat, melyek önállóan és általánosságban is a

kezelhető. Így például, le lehet kérdezni az összes tranzakcióra vonatkozó összesített adatokat, mint az összforgalom, fiókok összes terhelése. Másrészt igény lehet tranzakció típus specifikus lekérdezésekre, mint a pénzbefizetések vagy pénzfelvételek összege fiókonként egy magadott időszakra nézve. Ilyen közösen használt dimenziók esetében célszerű az egyes dimenziókat csak egyszer letárolni, s az összes változókat ezen közös dimenziókhoz irányítani. Az ilyen jellegű ténytáblákat, melyek egyazon dimenziókra épülnek kiterjesztett (extended) ténytábláknak szokás nevezni.

A közös dimenziók használatának előnye egyrészt a kisebb helyigény, másrészt a dimenzió értékek kezelésének konzisztensebb megvalósítása.




39. ábra, Kiterjesztett ténytáblák

A hatékonyság az adatszerkezet vagy adattartalom módosítási műveletek helyett elsősorban a lekérdezési műveleteknél bírnak alapvető fontossággal. Ugyanis a lekérdezési műveletek a módosítási műveletekkel ellentétben

- gyakran használatosak
- igen összetettek, sok számolási időt igényelnek
- a felhasználók elsődlegesen ezen műveleteken keresztül kommunikálnak a rendszerrel

Emiatt igen fontos, hogy az összetettebb lekérdezések is gyorsan hajtódjanak végre. Ehhez a DW rendszer az alábbi fontosabb optimalizálási lehetőségeket biztosítja:

- kapcsolatok közvetlen, pointeres mechanizmusa
- részeredmények letárolása, előaggregációk végrehajtása

A hagyományos relációs adatbázisokban a kapcsolat a különböző rekord előfordulások között két mezőcsoport értékegyezőségén alapszik. Vagyis ha például van egy autó és tulajdonos nyilvántartásunk és nyilván szeretnénk tartani, hogy a BZK651 rendszámú autónak a BGZ651F személyi igazolványszámú személy a tulajdonosa, akkor az autót leíró rekordba elhelyezünk egy olyan mezőt, amely tartalmazza a hivatkozott személy azonosító adatát, mint azt a 40. ábra is mutatja.

Ha a lekérdezési műveletek során az autó adatai mellé kapcsolni szeretnénk a tulajdonos adatait is, akkor az autót leíró rekordok mellé meg kell határozni a tulajdonost leíró rekord párjukat. Mivel az értékazonosságon kívül semmilyen más kapcsolat leíró elem nem szerepel az adatbázisban, ezért a rendszer csak a keresés módszerével tudja megtalálni a hivatkozott rekordot, vagyis például a BZK561 rendszámú autó tulajdonosának a meghatározásához az adatbázis kezelő rendszer motorjának meg kell keresnie a tulajdonos tábla rekordjai között azt a rekord előfordulást, amelyben az azonosító személyi igazolvány mező tartalma BGZ651F. Ugyan a keresést gyorsabbá lehet tenni, ha a közvetlen, szekvenciális keresés helyett egy indextábla alapján végezzük el a keresést, de ez a folyamat még így jóval is időigényesebb mint a közvetlen hivatkozás, s jóval több tárolóterületet is foglal le. Másrészt azt is figyelembe kell venni, hogy az indexek létrehozása és karbantartása is időigényes folyamat, ezért nem minden lehetséges társítási szemponthoz létezik egy megfelelő index az adatbázisban. Így egyes, ritkábban használatos kapcsolatok esetén ez a gyorsítási mechanizmus sem alkalmazható.

Autó

| Rendszám | Típus | Szín | Tulaj |
|---|---|---|---|
| R23 | Fiat | kék | 23 |
| R56 | Lada | zöld | 11 |
| R61 | Skoda | piros | 23 |

Tulajdonos

| Igsz | Név | Lakcím |
|---|---|---|
| 11 | Péter | Dorog |
| 13 | Anna | Miskolc |
| 23 | Pál | Eger |

40. ábra, Rekordkapcsolat megvalósítás a relációs modellben

A keresés lépései egy indexen alapuló keresési mechanizmusnál a következők:

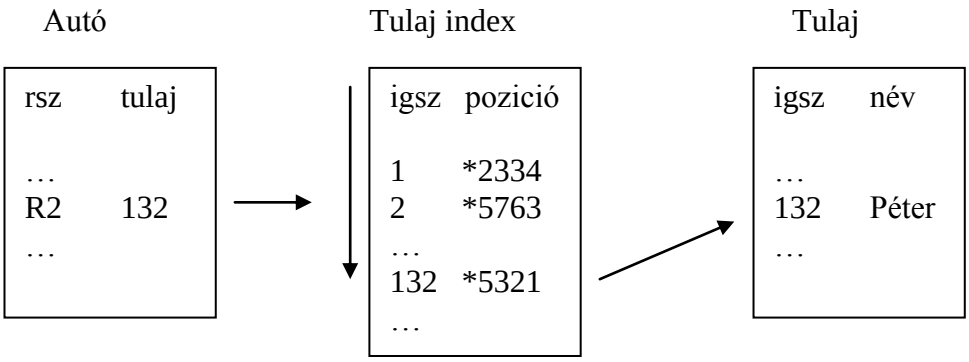
- kapcsoló mező tartalmának kiolvasása
- indexfa megnyitása
- a gyökér elemből kiindulva megkeresik az igényelt értéket
- a megfelelő indexbejegyzés megtalálása után ismert lesz a keresett rekord fizikai rekordcíme
- a megadott címről a rekordpár beolvasása

Ha a rendszerben nagyon sokszor kell kapcsolatokat kiépíteni a rekord előfordulások között, akkor a fenti asszociatív jellegű kapcsolat nyilvántartás nem tekinthető a legoptimálisabb megoldásnak. Hogy mégis ezt a módszert alkalmazzák a relációs adatbázisoknál, annak két fő érve lehet:

- a kapcsolat nyilvántartás ezen formája igen rugalmas, a kapcsolatok ad-hoc módon hozhatók létre a különböző rekord típusok között
- a relációs adatmodell az OLTP alkalmazásokra igazított, melyekben a lekérdezések rendszerint kevésbé bonyolultak és előre ismertek, így a

megtervezhető, hogy milyen indexekre lesz szükség az alkalmazás hatékony működtetéséhez.

Mivel az adattárház alkalmazások elsődlegesen az OLAP funkciók megvalósítására szolgálnak, itt sokkal több ad-hoc jellegű és összetett lekérdezés fog lefutni. Ezen lekérdezések egyik jellemzője, hogy több rekord típust is kapcsolatba kell hozni. Ezért az asszociatív jellegű társítás mechanizmusa nem a legmegfelelőbb mechanizmus. A kapcsolat nyilvántartásnak egy hatékonyabb módja az amikor a hivatkozó rekordban közvetlenül megtaláljuk a hivatkozott rekord fizikai címét, s nem kell ehhez egy indextáblát még külön igénybe venni. Vagyis ekkor egy pointer, egy mutató található a rekordban, amely megadja a társ rekord előfordulás fizikai címét.



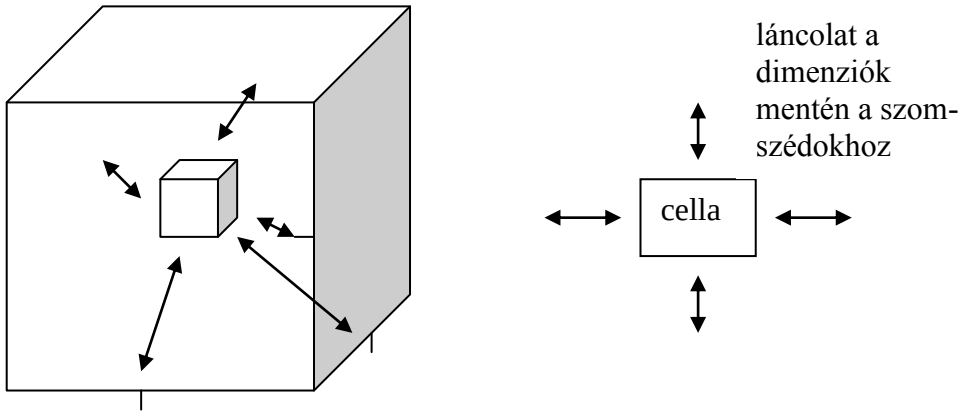

41. ábra, Kapcsolat keresés index alapján


41. ábra, Kapcsolat keresés pointer alapján

Egy ilyen pointereken alapuló keresésnél sokkal gyorsabban lehet meghatározni a kapcsolódó rekordpárt, hiszen nincs szükség külön indexhozzáférésre. A módszer hátránya viszont, hogy most minden támogatott kapcsolatnak be kell épülnie a struktúrába, hiszen ha egy rekord több más rekordhoz is kapcsolódik, akkor minden kapcsolódó rekord előforduláshoz léteznie kell egy megfelelő pointernek a rekordszerkezetben. Így egyrészt több kapcsolat esetén több pointert is tartalmaznia kell a rekordnak, másrészt újabb kapcsolat felvétele esetén módosítani kell a rekordszerkezetet is.

Mivel az adattárházakban egy adatkockát tekintve egy változó több különböző dimenzió értékhez is kapcsolódik, így a változó minden dimenzióhoz tartalmaz egy megfelelő pointert. Ezzel a pointerrel mutat a változót leíró rekordból a hozzá tartozó dimenzió rekordra. A lekérdezések kiértékelése során azonban nemcsak a változóból kiindulva válik szükségessé a dimenzió elérése, hanem sokszor a fordított irányú kapcsolatra is igény van. Így amikor egy dimenzió értékhez kívánjuk megadni a hozzá tartozó változó értéket, akkor a dimenzióból

lenne jó elérni a megfelelő adatcellát. S mivel több különböző adatkocka is osztozhat egyazon dimenzión, ezért egy dimenzió rekord több adatkockánk is a tagja lehet. Ezt azt is jelenti hogy a dimenzió rekordból több változó felé is kiindulhatnak pointerek. Vagyis az adatkockához egy egész pointer háló rendelhető, mely mindkét irányban összeköti a dimenziókat és a változó értékeket, az alapértékeket és a belőle származtatott értékeket. Az így kialakult pointer hálót szemlélteti az alábbi ábra.




42. ábra, Adatkocka pointer láncolata

*Előszámítások karbantartása*

Az OLAP rendszerek egyik fő jellemzője, hogy igen összetett lekérdezéseket kell végrehajtani, mely során nagy adatmennyiséget kell mozgatni, s több adatkockát kell érinteni, összetett és hosszú számításokat kell végezni. Ha például veszünk egy rendelés nyilvántartási OLAP rendszert, a felhasználó kiadhat egy olyan lekérdezést, melyben az elmúlt két hónapban megadott százaléknál nagyobb növekedést felmutató részlegek adatai jelennek meg válaszként. Tegyük fel, hogy az adatkocka az alábbi dimenzió szerkezetű:

- részleg (kód, név, város, cím)
- termék (kód, név, egységár)
- időpont (dátum)

Az adatkocka változójának szerkezete:

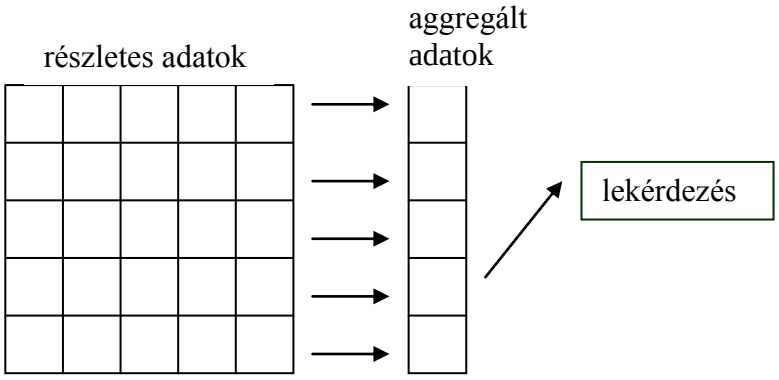
- mennyiség
- érték

Az igényelt lista előállításához az alábbi műveletek elvégzésére van szükség:

- az adatkockán egy ’drill up’ művelettel összevonjuk a különböző termékekre vonatkozó adatokat, így egy részleg-időpont szerinti összesített forgalom adatokat tartalmazó adatkockát kapunk.

- A kapott adatkockán egy újabb 'drill up' művelettel az idő dimenzió mentén végzünk összevonást, s napi adatok helyett havi összesítést tartalmazó adatkockát kapunk
- a részleg – hónap forgalmi adatokat tartalmazó adatkockából kiszámításra kerülnek a ez egyes havi különbségek megőrizve a részleg – hónap dimenzió struktúrát
- a kapott adatkockából megkeressük a szélsőértékeket, a legnagyobb növekedést mutató részlegek meghatározására

A felsorolás alapján látható, hogy különösen az első műveleti lépésekben igen nagy mennyiségű adatokon kellett összevonást, aggregációt végrehajtani.




43. ábra, Adatkocka előaggregáció

Ha gondolatban áttekintjük, hogy milyen jellegűek a gyakorlatban felmerülő lekérdezések, és elemezzük az eredmény előállításához szükséges elemi műveleteket, akkor azt tapasztaljuk, hogy nagyon gyakran van szükség az előzőekben is említett összesítésekre, mivel a felhasználók a részletes adatokat nem tudnák áttekinteni azok hatalmas mennyisége miatt. Ezért a lekérdezés eredményének előállítása tipikusan egy vagy több aggregációs lépéssel kezdődik.

A lekérdezés végrehajtásának gyorsítására ez a megfigyelés úgy hasznosítható, hogy így célszerű az igényelt aggregált értékeket tartalmazó adatkockákat előre kiszámítani, letárolni, s a műveletek végrehajtása során nem kell újra elvégezni az aggregációs számításokat, fel lehet használni az előszámítások eredményeit.

Az előszámítások alkalmazásával így jelentősen lecsökkenthetők a számítási műveletek végrehajtási ideje, ezért igen elterjed optimalizálási módszert jelentenek az előaggregációk. Az adattárház szempontjából nézve az alábbiakban foglalhatók össze e mechanizmus pozitív vonásai:

- gyorsabb művelet végrehajtás: mint már láttuk, az előszámítások letárolásával elhagyhatók bizonyos számítási műveletek a lekérdezések megválaszolása során, hiszen azok eredménye már rendelkezésre áll az adattárházban
- viszonylag kis extra helyigény: mivel az aggregált adatok mindig összesítést jelentenek, s az összesítés helyigénye sokkal kisebb mint az alapadatok halmazának helyigénye, ezért egy aggregált adatokat tartalmazó adatkocka sokkal

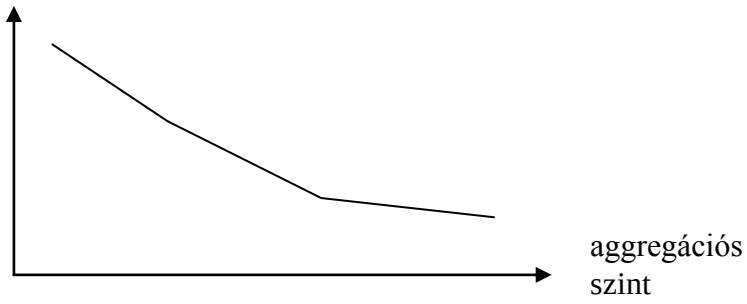
kevesebb tárhelyet foglal le, mint az alap adatkocka, az így jelentkező helyigény növekedés nem jelentős az alapadatok méretéhez képest

- transzparens a felhasználók számára: az aggregált adatkockák felhasználása, léte nem jelenik meg a felhasználók előtt, nem a felhasználónak kell kijelölnie, hogy a számítások során milyen előszámítási táblákat kell bevonni. A végrehajtó rendszer maga dönti el a rendelkezésre álló metaadatok alapján, hogy van-e olyan elő-aggregációs adatkocka amelyet fel tud használni a lekérdezés végrehajtásának gyorsítására van nem áll rendelkezésre semmilyen segédadat.

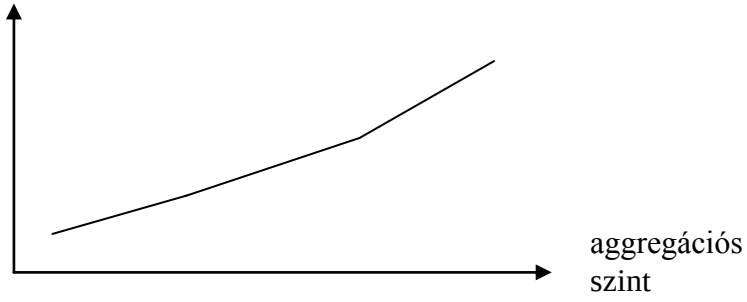
Ugyan mint említettük egy aggregált adatkocka helyigénye lényegesen kisebb, mint az alap adatkocka helyigénye, azonban a sok kicsi sokra megy elvét igazolva, a felmerülő helyigény többlet már jelentős lehet, ha igen sok előaggregációt kell letárolni a rendszerben. Ezért ugyan a lekérdezés szempontjából hasznos lenne, ha minden lehetséges aggregációs eredményt külön-külön letárolásra kerülne, azonban a tervezésnél figyelembe kell venni a helyigény növekedést is, ami korlátot szab a megvalósított aggregációs részeredmények halmazának.

A helyigény korlátok mellett egy másik fontos korlát a részeredmények letárolásánál az a tény, hogy az egyes előszámítások adatkockái nemcsak időnyereséget jelentenek, hanem bizony időigény növekedést is, hiszen az előaggregációk egy megadott időpontban készültek, s az adott pillanatbeli állapotot tükrözi, miközben az adatrendszer aktuális állapota módosulhat, s eltérhet az előállítási időpontbeli állapottól. Természetesen, ha egy ilyen régi, elavult állapotot használnánk fel az előszámítások gyorsítására, akkor téves, hibás eredmény adatokat kapnánk, ami nem megengedhető. Ezért gondoskodni kell arról, hogy az alapadatok módosításakor az aggregált adatok is módosuljanak. Mindez annyit jelent, hogy az alapadat rendszer értékének változásakor újra frissíteni kell az egyes aggregációs értékhalmazokat is. Ez viszont extra időigényt jelent.

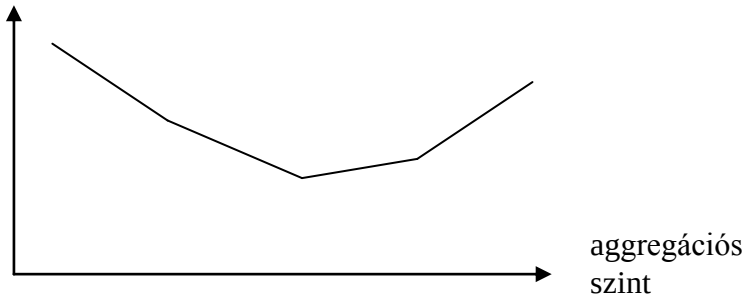
A következő ábrákon a megvalósított aggregációk mennyiségének és kapcsolódó költségeinek a kapcsolatát láthatjuk.




44. ábra, Lekérdezési költség



45. ábra, Módositási költség




46. ábra, Összesített költség

Az említett megfontolások is azt támasztja alá, hogy nem lehet minden lehetséges előszámítási aggregációt megvalósítani, szelektálni kell, hogy mely aggregációkat érdemes megvalósítani, s melyek lesznek, amelyek a lekérdezés során fognak dinamikusan létrejönni. Vegyünk például két lekérdezést a korábban említett forgalom adatkockára vonatkozólag, melyek a következőképen fogalmazódnak meg:

- az egyes részlegek adatai hónap bontásban
- az egyes hónapok összesített adatai

Ekkor a korábban említettek szerint itt több elő aggregációs részeredmény is szerepel a számításokban. Ha nem lehet mindkét művelet összes előszámítási igényét figyelembe venni, akkor a szükséges redukció egyik lehetséges megoldása, hogy csak az egyik lekérdezés igényeit vesszük figyelembe, s a másik lekérdezést összes adatának majd menet közben kell kiszámítódnia az alapadatokból. A preferált művelet kiválasztásában az egyes műveletek becsült gyakoriság adhat mérhető támpontot. Sajnos azonban a kiesett műveletet végrehajtók lényegesen nagyobb időköltséget lesznek kénytelenek elviselni.

Ezen igazságtalannak tekinthető optimalizálás mellett létezik szerencsére egy kiegyenlítettebb terhelést biztosító módszer is, melyben minden lehetséges művelet figyelembe vehető bizonyos mértékig. E módszer bemutatásához vegyük elő ismét az előző példánkat. Ha újra felírjuk a menet közben fellépő aggregációs adatkockákat, akkor a következő listát kapjuk:

- az egyes részlegek adatai hónap bontásban
  - idő szerinti aggregáció hónapokra
  - termék, idő szerinti aggregáció
- az egyes hónapok összesített adatai
  - idő szerinti aggregáció hónapokra
  - termék, idő szerinti aggregáció
  - részleg, termék, idő szerinti aggregáció

E listából jól látható, hogy a két művelet aggregációs részadatainak halmaza nem független egymástól, s az első művelethez használatos aggregációk a másodikban is megjelennek. Tehát egy aggregációs adatkocka több műveletben is szerepelhet. A másik fontos észrevétel, hogy az egyes aggregációs adatkockák sem függetlenek egymástól, hiszen egyes aggregációs adatkockákat ki lehet számítani más aggregációs adatkockákból, nem kell újra az alapadatokból kiindulva elvégezni minden egyes műveletet. Vagyis ha például szükség van egy

- termék, idő

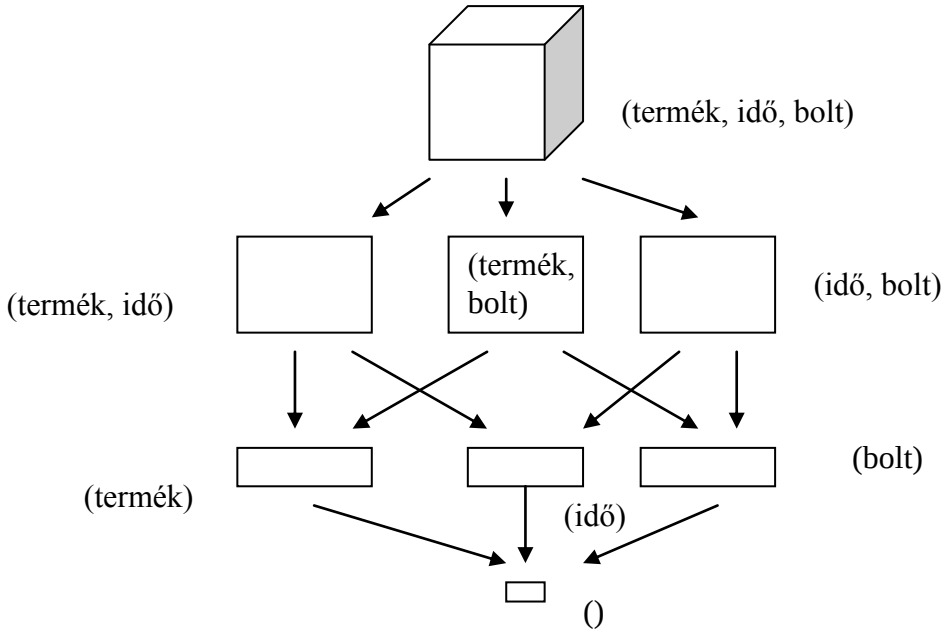
szerinti aggregációra, akkor az összesítéseket nemcsak az alapadatokból határozhatjuk meg, hanem a

- termék

vagy

- idő

szerinti aggregációkból is, hiszen ezekre egy második idő vagy termék szerinti aggregációt végrehajtva a kívánt eredményt kapjuk. Vagyis a különböző aggregációs adatkockák között egy leszármazási kapcsolat lehet, melyet egy irányított gráffal lehet reprezentálni, mint azt az alábbi ábra is mutatja.




47. ábra, Előaggregációk hálója

A fenti gráfban a csomópontok az egyes aggregációk, s egy $A \rightarrow B$ él azt mutatja, hogy a B aggregáció elkészíthető az A aggregáció ismeretében. Emiatt a gráfot leszármazási hálónak szokás nevezni. A háló elnevezés arra utal, hogy a gráfban bármely két elemnek van közös alsó szomszédja és közös felső szomszédja. A szomszédok jelentése a következő. A gráfban egy rendezési reláció értelmezhető az aggregációk között:

$A < B$

ha létezik $A \rightarrow B$ él, azaz B előállítható A-ból. A B aggregáció az A-nak felső szomszédja, ha a

$A < B$

és nem létezik olyan C aggregáció, melyre

$$A < C < B$$

teljesül. Hasonlóan értelmezhető az alsó szomszéd jelentése is. Két A,B aggregáció közös szomszédja olyan C aggregáció, amely nagyobb mindkettőnél és nincs olyan D aggregáció, amely szintén nagyobb mindkettőnél, de kisebb C-nél.

Az egyes aggregációkat azon dimenzió halmazzal reprezentálhatjuk, amelyek még megmaradtak az adatkockában egy induló adatkockát feltételezve. Ekkor minden aggregáció egy-egy dimenzió részhalmaznak felel meg. A fenti leszármazási háló alapján látható, hogy nem szükséges minden aggregációt letárolni, ha nincs elég hely, ehelyett csak bizonyos aggregációk foknak megvalósulni a többiek ezekből fognak leszármazódni igény esetén. A fenti mechanizmus egyik alapkérdése, hogy vajon mely aggregációk legyenek megvalósítva, s melyek készüljenek el menet közben az igény fellépte esetén.

Az optimális megvalósítandó aggregáció halmaz meghatározásának több különböző heurisztikus megoldási módszere terjedt el, mivel a feladathoz nem létezik egy hatékony egzakt módszer. A heurisztikus módszerek közül a legegyszerűbb a falánknak (greedy) nevezett módszer. A módszer onnan kapta az elnevezését, hogy amíg lehet újabb és újabb aggregációkat von be a megvalósítandó aggregációk körébe. Hogy mikor kell megállni a további bővítéssel, azt az algoritmust felhasználó illető határozza meg megadva egy költség függvényt és egy felső korlátot amit nem szabad túllépni. Egy lehetséges költségfüggvény lehet például a szükséges helyigény korlátként a maximálisan felhasználható tárolási hely nagyságát megadva. Egy másik költségfüggvény lehet a végrehajtási idővel arányos mennyiség, s a korlát egy megadott műveletcsomag végrehajtási idejéhez kapcsolódik.

A költségfüggvény és a korlát ismeretében az algoritmus mindig a legkedvezőbb jelöltet választja ki a megvalósítandó aggregációk halmazába. A módszer algoritmusa a következő:

```
S = {alapkocka}
while költség(S) < korlát do
        Válaszd ki A-t, melyre B(A,S) maximum
        S = S ∪ {A}
end
```

A fenti algoritmusban az egyes szimbólumok jelentése a következő:

- S : a megvalósított aggregációk halmaza
- A: egy aggregáció a még meg nem valósítottak közül
- B(A,S) : az A megvalósításával elérhető nyereség mértéke

A B(A,S) nyereség az A aggregációtól és az S aggregáció halmaztól függ. A nyereség jelentése azon felszabaduló költségek összege, melyek azáltal keletkeznek, hogy A-t bevesszük az S halmazba. A ciklusmagban azon A kerül kiválasztásra, melyre az így elérhető nyereség a legnagyobb értékű, így B nyereség kiszámításának menete a következő:

$$B(A,S) = \Sigma\, b(X,A,S)$$

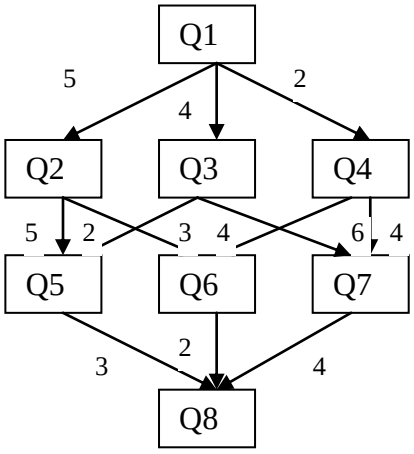
Ahol az összegzés azon X aggregációkra történik, melyek leszármaztathatók az A aggregációból. A b nyereség függvény egy X aggregáció előállítási költségekor fellépő

nyereséget adja meg, ha A megvalósított összehasonlítva azzal az esettel, amikor A nem lenne benne az S halmazban. A b() értéke nem lehet negatív, így mindig csak nem-negatív B értéket fogunk kapni.

$$B(X,A,S) = \begin{cases} c(X,S) - c(X,A), & \text{ha } c(X,A) > c(X,A) \\ 0 & \text{különben} \end{cases}$$

ahol c(X,A) az X aggregáció előállítási költsége az A aggregációból, illetve c(X,S) az X aggregáció előállítási költsége az S aggregáció halmazból. A módszer tehát minden lépésben összehasonlítja a lehetséges jelölteket egymással, s a legígéretesebb jelölt kerül kiválasztásra. A módszer ennek ellenére nem tudja minden esetben a globális optimumot szolgáltatni, hiszen egy jelölt kiválasztása hatással van az utána következő választások eredményeire, így az eredmény csak az adott választási sorrendre nézve optimális, azaz csak egy lokális optimum elérése biztosított.

A következőben vegyünk egy minta leszármazási hálót, melyben az élek mentén az egyes előállítási költségek láthatók.




48. ábra, Minta előaggregációs háló

Az algoritmus első lépcsője a S inicializálása az alapadattal:

S = { E1 }

Az első bővítési lépéshez ki kell számítani az egyes jelöltek jósági értékeit, melyek a hozzájuk tartozó nyereség értékekkel arányos a belőlük származtatható aggregációs táblákra vonatkoztatva. Az alábbi táblázatban megadtuk az egyes jelöltekhez tartozó elemi nyereségek és összesített nyereségek adatait.

| egyed | leszármazott | K1 | K2 | Nyereség |
|---|---|---|---|---|
| E2 | E5 | 7 | 5 | 2 |
| | E6 | 7 | 2 | 5 |
| | E8 | 9 | 4 | 5 |
| E3 | E5 | 7 | 3 | 4 |
| | E7 | 6 | 4 | 2 |
| | E8 | 9 | 6 | 3 |
| E4 | E6 | 7 | 6 | 1 |
| | E7 | 6 | 4 | 2 |

| | | | | |
|---|---|---|---|---|
| | E8 | 9 | 8 | 1 |
| E5 | E8 | 9 | 3 | 6 |
| E6 | E8 | 9 | 2 | 7 |
| E7 | E8 | 9 | 4 | 5 |

Az összesített nyereség az egyes jelölteknél:

E2: 12 E3: 9 E4: 4
E5: 6 E6: 7 E7: 5

Ezen értékek alapján a legtöbb nyereséget a E2 bevonása hozza, így ez az elem kerül másodikként felvételre a megvalósítandó előaggregációk listájába. A második lépés után tehát

S = { E1 , E2}

A harmadik lépésben a maradék elemekre végezzük el a kiválasztást az előzőekhez hasonló módon. Az elemi nyereségek listája:

| egyed | leszármazott | K1 | K2 | Nyereség |
|---|---|---|---|---|
| E3 | E5 | 5 | 3 | 2 |
| | E7 | 6 | 4 | 2 |
| | E8 | 4 | 6 | 0 |
| E4 | E6 | 2 | 6 | 0 |
| | E7 | 6 | 4 | 2 |
| | E8 | 4 | 8 | 0 |
| E5 | E8 | 4 | 3 | 1 |
| E6 | E8 | 4 | 2 | 2 |
| E7 | E8 | 4 | 4 | 0 |

Az összesített nyereség az egyes jelölteknél:

E3: 4 E4: 2
E5: 1 E6: 2 E7: 0

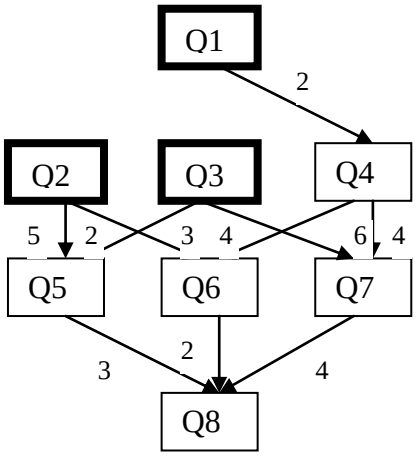
Így most az E3 kerül bevonásra, tehát az eredmény három lépés után

S = { E1 , E2, E3}

Az eredmény alapján fenti három aggregáció kerül megvalósításra, előszámításra az adattárház rendszerben, melyet az alábbi ábrával is szemléltetünk.

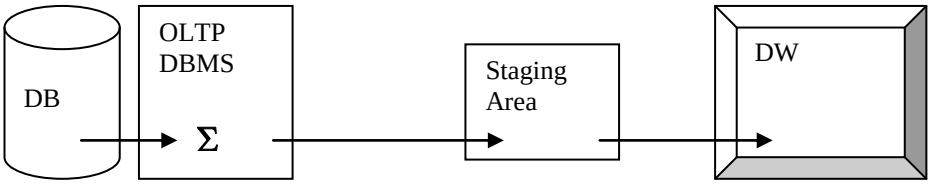
Az aggregáció kérdésében a teljesség kedvéért még egy vetületet szokás figyelembe venni, nevezetesen azt, , hogy mikor kerül sor a megvalósított aggregációk frissítésére. Ugyan az nyilvánvaló, hogy az adattárház alapadatainak a módosítása után kell elvégezni a frissítést, de hogy pontosan mely részlépésben történjen arra több különböző alternatíva is létezik:

- a forrás adatbázisban
- az átemelési területen
- a betöltés alatt
- a betöltés után



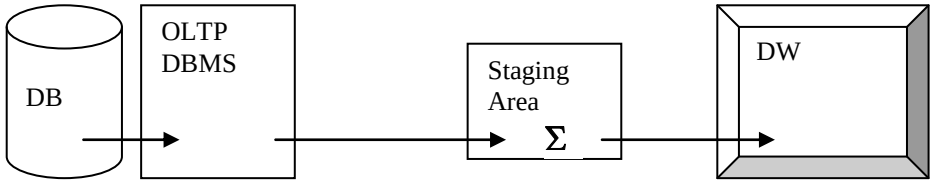

49. ábra, Minta előaggregáció megvalósítási háló

Az első változatban már magában a forrás adatrendszerben elvégzésre kerülnek az összesítések, minden aggregáció más és más forrásban számítódhat ki. Az átemelés során az aggregált adatok az alapadatokkal együtt kerülnek át az adattárházba. E megközelítés előnye, hogy tehermentesíti az adattárházat az extra frissítési műveletek alól. Viszont a módszer nagy hátránya, hogy a forrás adatbázist kell úgy módosítani, hogy ott létrejöjjön az aggregáció, másrészt csak az egyazon forrásból származó adatok esetén lehetséges az aggregáció elvégzés, a különböző forrásokból származó adatok összesítése továbbra is csak az adattárházban lehetséges.



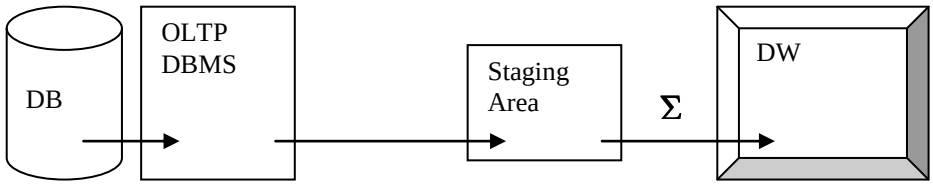

50. ábra, Frissítés a forrás adatbázisban

A második változatban az átemelési területen, az adatoknak a közös tárolási formába való átültetése után történik meg az összesítés. Ebben az esetben is igaz, hogy az aggregáció még az adattárházon kívül mehet végbe, csökkentve az adattárház belső karbantartási leterhelését. A másik előny, hogy itt már rendelkezésre állnak a több különböző forrásból származó adatok is, így szélesebb lehet az aggregációba bevont adatok köre. Igaz viszont az is, hogy az aggregációhoz most is csak az új adatok állnak rendelkezésre. Az adattárházban már korábban letárolt adatokkal csak a későbbi fázisokban lehet elvégezni az összesítést.



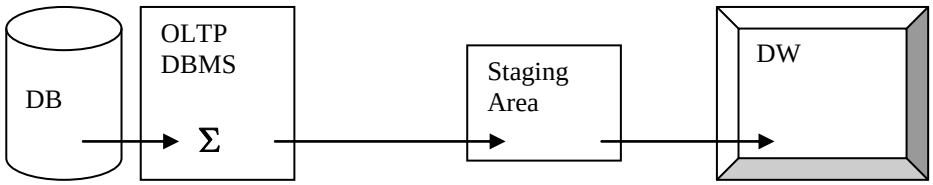

51. ábra, Frissítés az átemelési területen

A harmadik változat esetében már minden adatelem rendelkezésre áll a teljes aggregáció elvégzésére, s így minden lehetséges összesítés frissíthető. Viszont itt a frissítés a betöltés részeként megy végbe, s ezt már az adattárház működési költségét növeli. Az ekkor elvégzett frissítések lassítják a betöltés, s a tárház funkciók folyamatát.




52. ábra, Frissítés a betöltés során

A negyedik változat esetén a betöltés és a frissítés folyamata szétválik. Itt is igaz, hogy minden adatelem rendelkezésre áll a teljes aggregáció elvégzésére, de itt a frissítés a betöltés után megy végbe. Ugyan ez is az adattárház működési költségét növel, de ez a betöltési leterhelés után is végbe mehet, amikor a rendszer leterhelése már kisebb. Ez a késleltetés a teljesítmény szempontjából előnyös, viszont az alapadat betöltés és a frissítés közötti időben a rendszer alap és aggregált adatai nem konzisztensek egymással.




53. ábra, Frissítés a betöltés után

A hatékonysági tulajdonságok áttekintése után a következő részben az adatok helyességének kérdését vesszük elő. Ez a rész is lényeges a DW rendszer működése szempontjából, hiszen tudjuk, hogyha szemét adatok kerülnek be a rendszerbe, akkor csak szemét adatok jöhetnek ki belőle.

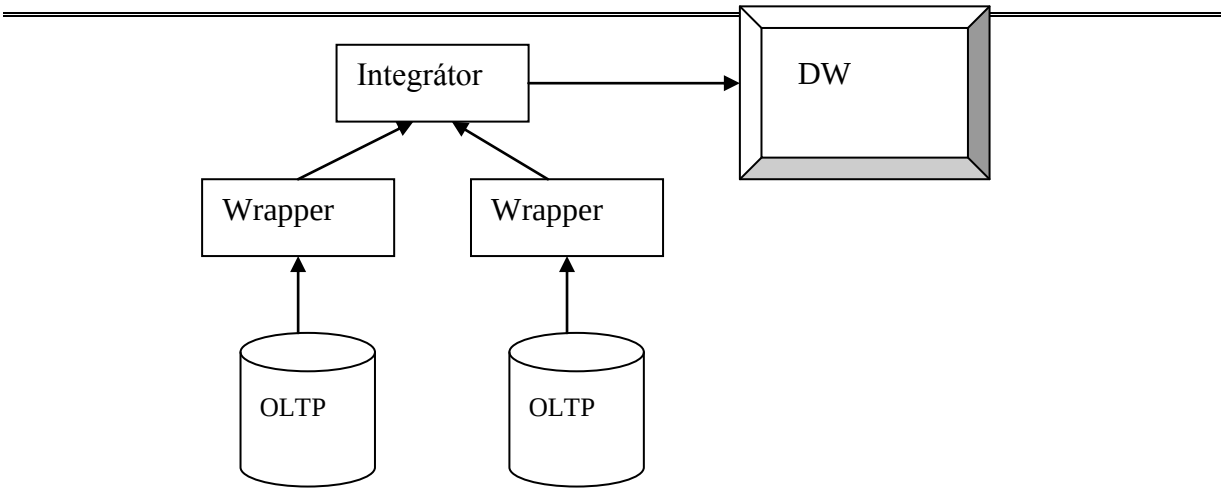
**Betöltési folyamatok, adattisztítás**

Az adatbetöltés folyamata egyik legfontosabb esemény az adattárház működése során, hiszen ekkor kerülnek be adatok a külső forrásokból, s mint már láttuk ekkor nemcsak egyszerűen bejönnek az adatok, hanem beilleszkednek a már meglévő adatok közé. Ezen beilleszkedési művelet igen komplex tevékenység is lehet, hiszen a betöltés során gondoskodni kell arról, hogy

- homogenizálni kell az inhomogén forrásokból bejövő adatokat
- konzisztensé kell tenni a különböző forrásokból származó, egymással inkonzisztens adatokat
- tisztítani kell az adatokat a rossz, téves, bizonytalan értékek esetén
- be kell tölteni az adatokat fizikailag is az adatkockákba
- aktualizálni kell a kapcsolat leíró pointereket
- frissíteni kell a megvalósított aggregációs adatkockákat

Ugyan nem tartozik szorosan az adattisztítás témaköréhez, de itt, az adatok dinamikus jellegének vizsgálatakor lehet megemlíteni azt is, hogy az adattárházak esetében nemcsak az adatbetöltés az egyedüli dinamikus esemény, hanem az adatok kiürítése is. Ezen műveletre azért van szükség, mert az adattárházak mérete sem végtelen, a rendelkezésre álló tárolóhely kapacitás véges. Ha mindig csak bevinnénk adatokat, akkor egy idő után elérkeznénk e méretkorláthoz, s nem tudnánk további adatokat felvinni. Ezért sok esetben célszerű a már lejárt érvényességű adatokat kiüríteni, kivenni a rendszerből, hogy helyet kapjunk újabb, frissebb adatok számára. Természetesen annak eldöntése, hogy mikor és mit szabad kivenni a rendszerből komolyabb előkészítést igényel. Mivel e méretkorlátok igen tágak lehetnek és sokszor nem jár le egy adatelem érvényessége igen hosszú ideig, a gyakorlati rendszerekben ritkábban van szükség ilyen jellegű adat kiürítésre, sokkal inkább fontosabb az adatbetöltés folyamata, mely egy rendszeresen lezajló tevékenység.

A betöltési folyamat összetettsége és széles lehetőségei miatt a gyakorlati rendszerekben különböző mértékben valósulnak meg az egyes betöltési modulok. Az egyszerűbb adattárházakban például nem szerepelnek az egyes adattisztítási funkciók. Ezen különbségek miatt a betöltési rendszer struktúráját általánosabb szinten érdemes felvenni, melyben nem annyira az egyes funkciók típusai, hanem inkább a lényegesebb funkció csoportok kaptak helyet. Egy ilyen általánosabb sémát mutat be a következő ábra.




54. ábra, Betöltési modul struktúrája

A modellben a wrapper komponens végzi az egyedi adatforrásból a közös formátumra történő konverziót. A monitor, figyelő komponens célja az áttöltési folyamat elindítása, ha lényeges változás történt az alatta elhelyezkedő adatforrásban. Ha a monitor program változást észlel a hozzá tartozó forrás adatrendszerben, akkor elindítja a betöltési folyamatot. A kiemelt adatok az integrátor részhez kerülnek, melyben a különböző helyről jövő, tartalmilag heterogén adatokat beépíti a logikailag egységes adattárházba. Ennek során többek között szűri, illeszti, tisztítja a bejövő adatokat, s elvégzi az adatoknak a fizikai beépítését a meglévő adatstruktúrába.

Az adatbetöltés folyamata négy alapvető résztevékenységre bontható:

- adatok kiolvasása az adatforrásból (data extracting)
- adat tisztítás (data cleaning)
- adat konzisztencia ellenőrzés (view consistency maintenance)
- adatok tényleges beépítése az adattárházba (physical data loading)

Az adatok kiolvasása során a forrás adatrendszerből kikerülnek az adatok egy közbenső tárterületre és ott egy közös adatformátumra konvertálódnak, hogy össze lehessen illeszteni a különböző forrásból származó adatelemeket. Az adatkiolvasás folyamata magába foglalja

- az átemelendő adatok kijelölését a forrás adat rendszerekben. Ehhez meg kell határozni, hogy mely adatelemek épülnek be az adattárházba, s ezek közül melyek értéke változott az utolsó átemelés óta.
- A kiolvasott adatok közös formátumra alakítását, hogy el lehessen végezni a későbbi lépésekben a különböző helyről származó adatok illesztését, összevonását.

Az adatkiolvasás egyik fő jellemzője, hogy mi aktivizálja ezt a folyamatot, mikor kezdődik el a kiolvasás művelete. Egyrészt az adattárház indulásakor kell végrehajtódnia, hogy létrejöjjön az adattárház induló adatrendszere, másrészt a későbbi üzemszerű működés során is szükség van átemelése, amikor frissítésre kerül az adattárház tartalma. Az adatok kiemelésének és az adattárházba való megismétlésének műveletét szokás adat replikációnak is nevezni. A replikáció művelete lehet szinkron és aszinkron. A szinkron replikáció esetén a forrás rendszerben történő módosítási művelet, például egy SQL UPDATE utasítás váltja ki az adatátemelés elindítását. Az adatmódosítási művelet egyrészt maga triggerelheti az adatátemelő funkció indulását, másrészt beállíthat egy jelzőt, melyet detektálva a rendszeresen feléledő átemelő folyamat felügyelő rutin észrevesz és aktivizálja az átemelés folyamatát.
Aszinkron replikáció esetén az átemelés folyamata az adatérték módosulását követően indul csak el. Ennek elsősorban hatékonysági szempontból van előnye, hisz így több módosulás nem egyenként, hanem együtt, egy replikációs folyamatban kerülhet át az adattárházba. Aszinkron üzemmód esetén a módosítandó értékek meghatározása történhet

- adatlista segítségével, amikor a módosítási műveletek maguk teszik ki a módosított értékeket egy külön állományba, illetve
- tranzakció napló segítségével. Ebben az esetben a DBMS által rendszeresen készített tranzakció napló alapján tudja a DW rendszer megállapítani, hogy mely adatelemek is módosultak az utolsó átemelés óta.

Az adattisztítási műveletek fő célja, hogy biztosítsa az adattárház modell egységességét, integritását. Az adattisztítási modul megszűri a bejövő adatrendszert az esetleges ellentmondásoktól, anomáliáktól. Ellentmondás jelentkezhet a különböző helyről bejövő adatelemek között például a mező hossza, a mező adattípusa vagy a mező elnevezése között. Igen sok problémát jelenthetnek az opcionális vagy hiányzó értékű mezők esetei is. Mivel az adatértékeket érintő ellentmondásoknak több különböző típusa létezik, így feloldásukra is több különböző módszert lehet alkalmazni.

Az adattisztítás legegyszerűbb formája az adatmigráció. Ez a folyamat egy egyszerű transzformációt takar. A transzformációs szabályok a helyettesítési formulák megadásával írhatók le. Ez a módszer akkor előnyös, ha a felhasználó az adatelemek egyszerű helyettesítésével meg tudja oldani a konzisztencia szint javítását. Ha például a dolgozók nemének jelölésére az egyik adatforrásban a Férfi, Nő értékeket használják, míg a másik adatforrásban F, N szimbólumok terjedtek el, akkor az egységes adatrendszerhez szükség van a különböző adatérték formátumok egységesítésére. Így például előírható egy adatmigrációs szabály keretében, hogy minden Férfi érték konvertálódjon F értékre, és minden Nő érték alakja pedig N legyen.

A másik elterjedt adattisztítási művelet az adatsúrolás (scrubbing) folyamata, mely során alkalmazási terület specifikus konzisztencia ellenőrzéseket hajtanak végre. Ebben a fázisban már nemcsak tartalom nélküli karakter sorozatként tudják az adatértékeket kezelni, hanem már jelentés is társítható az egyes adatelemekhez. A lehetséges ellenőrzési mechanizmusok közül az egyik legfontosabb a mezők feldarabolása, s az egyes tagok jelentésének meghatározása átrendezése, kijavítása. Egy másik gyakori feladat a különböző helyről származó rekordok illesztése, mely során az egyes rekordok tartalma határozza meg az illeszkedési feltételt. Ezen fázis jellemzője, hogy a tervező maga határozza meg, hogy hogyan lehet tagolni a bejövő adatsort, s milyen szabályszerűségeknek kell teljesülniük az adott témakör bemenő adatainak.

Egy harmadik elterjedt módszer az adat auditáció, melynél már egy összetettebb szabály feltáró algoritmust alkalmaznak, hogy fényt derítsenek olyan rejtett összefüggésekre, szabályokra, melyek a megadott bemenő adatrendszert jellemzik. A szabályok feltárásához különböző adatbányászási módszereket lehet felhasználni. Ezt követően ellenőrzésre kerül, mely bemenő rekordok azok, amelyek nem teljesítik az így feltárt szabályszerűségeket. Ha az ellenőrzés a szabályoktól eltérő értékkel bíró bemenő adatokra bukkan, akkor ez az adat kikerül egy külön listába, s rendszerint egy üzenet is generálódik a kezelő személyzet felé megadva a normálistól eltérő adatokat. Az előző adattisztítási módszertől eltérően itt maga a betöltési modul határozza meg, hogy milyen szabályszerűségek jellemzik a bemenő adatsort, s nem külső paraméterként kerül megadásra az ellenőrzésre kerülő szabályrendszer.

Az adattisztítás említett műveleteinek elvégzése bizonyos előkészítési munkákat igényel, hiszen egy struktúrálatlan szövegből, értékhalmazból reménytelen feladat a helyes tartalom kiemelése. Ha például veszünk egy

Pál Nagy dr Dorog 524 ut Kalman Janos 676

elérési címnek megadott karaktersorozatot, nem lehet egyértelműen eldönteni, hogy a szöveglánc egyes tagjai milyen jelentéssel is bírnak. Az értelmes átalakításokhoz előbb elő kell készíteni a adatokat. Az adatelőkészítés folyamata rendszerint az alábbi lépésekből áll:

- szöveglánc darabokra bontása, az értelmes szövegdarabok elhatárolása
- egyes szövegdarabok funkcióinak, jelentésének beazonosítása
- egyes szövegelemek szabványos alakra konvertálása

A fenti kiinduló szövegláncból például az alábbi struktúrált információ alakulhat ki az adattisztítás előkészítése során:

- családi név: Nagy
- keresztnév: Pál
- rang: Dr
- város: Dorog
- irányitószám: 524
- utca : Kalman Janos
- tipus: ut
- hazszam: 676

A betöltés folyamatához kapcsolható az előző fejezetben említett elő aggregációs optimalizálási mechanizmus is, hiszen az aggregációk frissítésének egyik lehetséges változata, amikor a betöltés során kerülnek frissítésre az aggregált adatkockák. Az aggregáció aktualizáló modul egyik feladata, hogy meghatározza, egy adott típusú bejövő adatelemnél mely aggregációkat kell módosítani, s hogyan végezhető el a leggyorsabban a szükségessé váló frissítés. Egy érdekes és sokat tanulmányozott vetülete az aggregáció frissítésnek az a tény, hogy amíg az adattárház adatkockák adatai több különböző adatforrásból származnak, s az aggregációk is több különböző forrásból származó adatokon nyugszanak, addig a különböző adatforrásokból származó adatok eltérő időben kerülhetnek frissítésre. Így előfordulhat, hogy az aggregációba bevont adatok egy része egészen más időponthoz kapcsolódik, mint egy másik forrásból bejövő része, hiszen a különböző források eltérő időben aktualizálódhatnak. Emiatt az aggregációk, s maguk az adatkockák adatai is bizonyos értelemben inkonzisztensek, hiszen nem egy egyidejű pillanatfelvételhez kapcsolódnak.

A fizikai adatbetöltés folyamán kerülnek az adattárház struktúrába a már átkonvertált, megtisztított adatok  Ezen lépés megoldásánál a legnagyobb megoldandó probléma a hatékonyság kérdése, hogy mely módszerekkel és tárolási struktúrákkal lehet a leggyorsabban megvalósítani a fizikai adatbetöltés folyamatát. A fizikai betöltés is egy összetett tevékenység, amely az alábbi komponenseket tartalmazza:

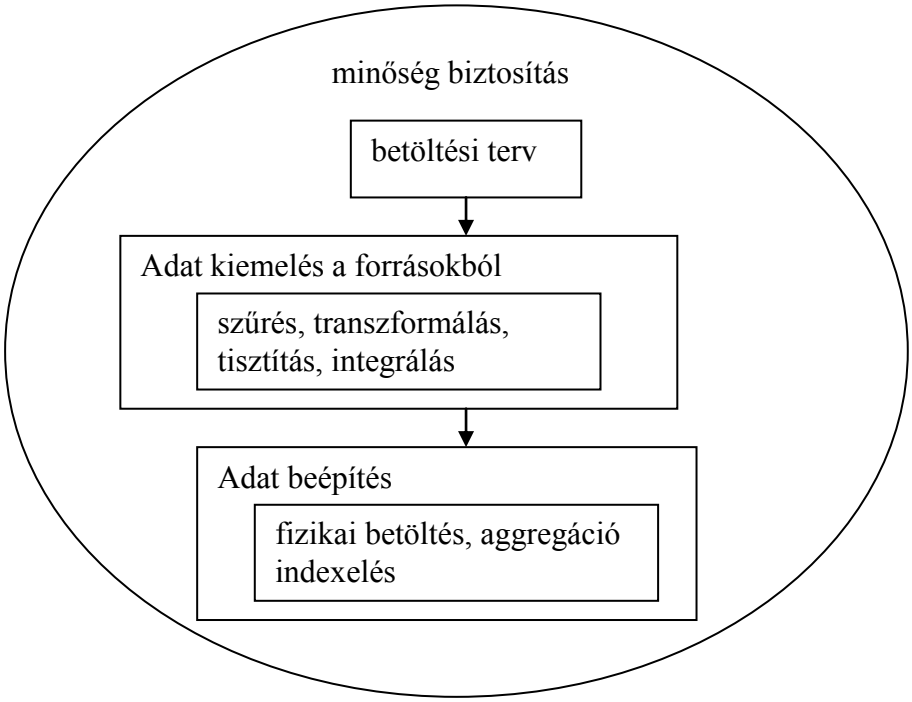
- a DW rendszer integritási eleminek ellenőrzése
- adatok rendezése
- indexek és pointerek létrehozása
- elő aggregációk módosítása
- adatok particionálása

A hatékonyság javítására a legígéretesebb megoldások a műveletek párhuzamosításához kapcsolódnak.

Habár az adatbetöltés folyamata több elemi részlépésből áll, mint az előzőekben be is mutattuk, mégis egységként célszerű kezelni a betöltés műveletét, hiszen ezen részlépések igen szorosan kapcsolódnak egymáshoz, az egyik által előállított adathalmaz felhasználásra kerül a következő műveletben. Így egy komplex konverziós tervet szokás kidolgozni az adattárház adatbetöltési moduljához, mely a teljes adatbetöltési folyamatot lefedi. A feladat sokrétűsége miatt e terv elkészítése több szakember együttes munkájának az eredménye. A konverziós terv kidolgozása során az alábbi tevékenységeket kell elvégezni:

- a cél adattárház struktúrájának és szemantikájának elemzése
- az adatforrás adatok struktúrájának és jelentésének elemzése
- adatelemek leképzésnek megtervezése, azaz annak meghatározása, hogy az egyes forrás adatok mely adattárház objektumokhoz kapcsolódnak
- a külső adatforrások elérési módszereinek meghatározása, itt megadásra kerül többek között az adatforrás helye, adatformátum, s az adat kiemelést végző program
- a szükségessé váló konverziós rutinok megtervezése
- az adatelemek szükségessé váló tisztítási műveleteinek a meghatározása
- adattárház rendszer minőségbiztosítási követelményeinek a biztosítása
- a tényleges adattisztítást és fizikai adatbetöltést végző szoftver elemek kiválasztása

A minőségbiztosítási lépések célja, hogy az adattárház a frissítés után is használható, konzisztens, tiszta legyen. A minőségbiztosítás szempontjai általános irányelvként működve áthatják a teljes adatbetöltési folyamatot.




55. ábra, DW rendszer minőség biztosítási elemei

A adattárház rendszerek minőség ellenőrzése a helyes és tiszta adatok biztosítására irányul. A minőségbiztosítási rendszer kidolgozása során az alábbi fontosabb lépésekből áll össze:

- a felhasználói igények összegyűjtése, milyen jellegűek az adatokkal szemben támasztott követelmények, mikor tekinthető az adatrendszer megfelelő minőségűnek.
- Módszerek kidolgozása a minőség romlását okozó jelenségek észlelésére, hatásuk mérésére
- a minőség romlását okozó tényezők meghatározása, s a negatív hatással járó folyamatok elkerülésére alkalmas módszerek, eljárások kidolgozása. A minőség biztosítását célzó lépéseket minél közelebb kell hozni az adatforrásokhoz, hogy a negatív hatásokat minél szűkebb területre lehessen beszorítani.
- Ki kell dolgozni a megfelelő validációs, ellenőrzési eljárásokat, melyek célja a minőségbiztosítási lépések szükséges színvonalon tartása-

Az adatrendszer minőségének értékelése során az alábbi fontosabb kulcs szempontokra célszerű kitérni:

- adatok relevanciája: az adatrendszerben letárolt értékek illeszkednek a felhasználók által igényelt adatelemekkel

- teljesség: minden a modellbe felvett adatelemhez tartozik egy érték, az adatelemek nem állnak üresen
- konzisztencia: a különböző helyről származó, azonos objektumokhoz tartozó értékek egyezőségének mértéke
- megbízhatóság: az adatrendszerben tárolt értékek megegyeznek a valóságban megjelenő értékekkel
- adatok rendelkezésre állása: a DW rendszer szükség esetén rendelkezésre áll a felhasználóknak, nem kell azoknak várakozniuk rendszer leterhelés vagy hiba miatt
- adatelemek egyedisége: a felvitt adatelemek azonosíthatósága, minden felvitt adatelemhez létezik megfelelő azonosító
- adatok ellenőrzöttsége: a felvitel során az adat értékek szűrésre és konverzióra kerülnek a téves, hibás értékek megszüntetése céljából

A minőségbiztosítás keretében a tervezés során a harmadik pont a minőség romlását okozó tevékenységek és jelenségek feltárása volt. Az elemzés során összesíteni kell a tényleges eseményeket és meg kell adni, hogy pontosan milyen veszélyeket rejt ez a esemény. Így például a rendelkezésre állás vetületét vizsgálva, egyik lehetséges veszélyforrás a rendszer összeomlása. Ezen esemény hatására a rendszer megadott időtartamig, a helyreállítás befejezéséig, nem engedélyezi a külső hozzáférést. Az események és hatásuk ismeretében kell összeállítani az események megakadályozására vagy hatásuk csökkentésére irányuló modulok kidolgozását.

## Az OLAP és DW rendszerek alkalmazási területei

A következő részben áttekintést adunk az adattárház rendszerek főbb alkalmazási területeiről, az alkalmazáshoz kapcsolódó lehetőségekről, s nehézségekről is. E fejezet célja hogy egyrészt segítséget nyújtson a DW rendszerek tervezése során, hogy meg tudjuk előre határozni milyen szerepet is fog a DW rendszer játszani majd a vállalat életében, mely feladatok elvégzésben lehet majd hasznosítani a kifejlesztett rendszert. A lehetséges alkalmazási területek mellett a felmerülő implementációs nehézségekre is szeretnénk itt rávilágítani, hogy a megmutassuk, mik azok a kulcsproblémák, amelyekre oda kell figyelni a rendszer megtervezése és implementációja során, hogy elkerüljük a mások által már megtapasztalt buktatókat, kudarcokat.

A DW rendszerek egyik fő alkalmazási területe a döntéstámogatási (DSS) feladatok elvégzése. A DSS rendszerekben a felhasználó összesített, trend adatokat, szabályszerűségeket, vagy a szabálytól eltérő viselkedést bemutató adatokat kérdezhet le felhasználó barát megjelenítésben, jelentések vagy grafikonok formájában. A kapott adatokat, szabályszerűségeket a döntéshozók figyelembe vehetik a döntéseik meghozatalánál. Természetesen azonban nemcsak nagy stratégiai, a vállalt életét alapvetően befolyásoló döntések esetében lehet alkalmazni a DSS eszközök által nyújtott lehetőségeket, hanem a mindennapi rutin feladatok során is hasznos lehet egy ilyen segédeszköz. A következő listában ezen tipikus rutin jellegű tevékenységekről adunk egy rövid áttekintést:

- általános, átfogó információk gyűjtése a vállalat jelenlegi állapotáról, annak ellenőrizése, hogy a terveknek megfelelően mennek a dolgok

- speciális, szelektált információk gyűjtése a vállalat valamelyik egységéről, annak ellenőrzése, hogy az adott egységnél vagy ügyfélnél a terveknek megfelelően mennek-e a dolgok
- a vállalat vagy egy részleg mutatóinak időbeli követése, a megadott paramétert megadott gyakorisággal lekérdezve képet kaphatunk a változások jellegéről, irányáról.
- egy megadott trend vagy jelenség alátámasztása, a DSS rendszer felhasználható arra, hogy a megérzésen alapuló trendeket, jelenségeket tényadatokkal is alá lehessen támasztani
- a kialakuló trendek, változási irányok minél rövidebb idő alatti felderítése
- a szélsőséges paraméterekkel rendelkező egységek, objektumok felderítése
- a szokványos működési rendtől eltérő egységek, ügyfelek, egyedek felderítése
- a vállalat teljesítmény adatainak bemutatása
- ad-hoc jellegű lekérdezések, jelentések gyors végrehajtása, anélkül, hogy ehhez egy programfejlesztő segítségét igénybe kellene venni. Erre a szolgáltatásra az ad lehetőséget, hogy az OLAP rendszerek igen rugalmas és felhasználóbarát kezelő felülettel rendelkeznek, melyekkel gyorsan lehet egyedi információ igényekre is eredményt előállítani.
- a jövőbeli állapotok előrejelzése, a DSS eszközökkel a változási trendek feltárhatók, s az aktuális állapotból kiindulva megbecsülhetők a jövőben megvalósuló állapotok
- különböző döntési változatok hatásainak összehasonlítása, egyes DSS eszközök rendelkeznek feltételes előrejelzési segédeszközökkel is, melyek segítségével az egyes döntési alternatíváknak a jövőbeli állapotokra kifejtett hatása is vizsgálható, elemezhető.
- a meghozott döntések hatásainak összegyűjtése. a korábban meghozott döntések után kialakuló paraméterek, trendek felfedésével rövidebb időn belül igazolható a döntés jogossága, vagy a döntés módosításának szükségessége.

A rutinszerű alkalmazások mellett néha azonban szükség lehet a stratégiai jellegű döntések elősegítésében is a DSS rendszerekre. A stratégiai döntések meghozatalát támogató rendszerek egyik jellemzője, hogy a létrehozott rendszer viszonylag rövid életű összehasonlítva a normál, szokásos DSS alkalmazásokkal. A döntések meghozatala ugyan nem percek vagy órák alatt történik, hanem napok, hetek alatt, de az az idő még így is sokkal rövidebb, mint a normál alkalmazások éves életciklusa. A másik fontos jellemző, hogy maguk a döntések igen komoly hatásúak, ezért igen sok oldalról meg kell vizsgálni őket, több lehetséges változat hatását is figyelembe kell venni. Ehhez egy jól előkészített DSS alkalmazásra van szükség. tehát a stratégiai döntések esetében igen hosszú előkészítési idővel és viszonylag rövidebb felhasználási idővel lehet számolni. Azt viszont nem árt tudni, hogy ezen rövidebb idő alatt több értéket termelhet a vállalat számára, mint az összes többi rutin jellegű alkalmazás együttvéve.

Mivel a stratégai döntések esetében a nagyon rövid idejű, de intenzív felhasználással lehet számolni a többboldalú vizsgálatok miatt, igen fontos kérdés a rendszer hatékonyságának a kérdése, hogy milyen gyorsan tudja a felmerő lekérdezéseket megválaszolni. A hagyományos üzemi DW rendszer alkalmazása során azonban komoly gondot jelent az, hogy egyrészt a normál DW-ben más lekérdezések is futhatnak a döntés előkészítési munkával párhuzamosan, másrészt azzal is számolni kell hogy a normál DW mérete igen terebélyes lehet, hiszen a vállalat teljes egészére kiterjedő történeti adatokat tárol. Nagyobb adathalmazok esetében

pedig lassabb lehet a válaszadási idő is. Emiatt célszerű lehet egy külön adatbázis, DW létrehozása a döntés támogatási feladatokhoz.

A külön adatbázis létrehozásának az is az előnye a kisebb méret mellett, hogy a műveletek végrehajtását a döntés támogatás specifikus lekérdezésekhez lehet igazítani. Ugyanis a normál DW rendszerekben a rutin jellegű feladatok dominálnak, s ehhez kapcsolódóan lettek kialakítva a pointerek, elő aggregációs adatkockák. Az igényelt feladat specifikus lekérdezések számba vételével ebben a külön adatbázisban már testre szabott optimalizálást lehet megvalósítani, ami még gyorsabbá teszi a végrehajtás műveletét.

A kialakított rendszer további lényeges jellemzője, hogy itt a felhasználók nem annyira a részletes adatokra, hanem inkább az erősen aggregált adatokra kíváncsiak. A stratégiai döntések ugyanis vállalati szintűek, ezért a hatásukat is vállalti, összesített szinten kell vizsgálni, s ehhez sokkal jobb áttekintést adnak az aggregált adatokat tartalmazó listák, mint az egyedi értékek felsorolása. Emiatt megengedhető az is, hogy az egyedi adatok kevésbé tiszták legyenek, ha az eltérést a többi elem hatása elnyomja, az aggregált értékekben nem okoz jelentős változást. A felhasználói felület tervezése oldalról nézve viszont többlet munkát jelent az, hogy itt jobban oda kell figyelni arra, hogy ezen eredmény adatokat olyan döntéshozó személyek fogják kézhez kapni, akik különben ritkábban kerülnek gép közelbe, ezért számunkra emberibb formában, grafikusan kell megjeleníteni az adatokat.

A stratégiai döntések meghozatalára szolgáló DW rendszerek kiépítésében valószínűleg több külső konzultánst is be kell vonni, hiszen a megoldandó feladat igen sokrétű és széles informatikai területről igényel ismereteket. A külső szakemberek esetében viszont ügyelni kell arra, hogy a kidolgozott megoldásaik, ismereteik minél nagyobb mértékben kerüljenek át a saját tudásbázisba. Ehhez következetesen meg kell követelni az elvégzett műveletek dokumentálását, az egyes lépések háttéranyagainak összegyűjtését. A DW fejlesztés vezetőjének azonban nemcsak az informatikai szakemberekre kell ügyelni, hanem a felhasználókkal is ki kell alakítania a megfelelő kapcsolatot. Nem szabad a felhasználókban azt a hitet kelteni, hogy a DW – DSS rendszer egymaga képes a helyes döntések meghozatalára. A DSS rendszerek csak segédeszközök, hogy a döntéshozót rövid idő alatt az igényelt információk birtokába jutassa. A DW rendszerek lehetőségire vonatkozó reális kép kialakulásában sokat segít az, ha a döntéshozókat minél korábban be tudjuk vonni a DW rendszer fejlesztési munkáiba.

Az OLAP rendszerek felhasználási területeire rátérve lényeges kihangsúlyozni, hogy az OLAP rendszereket igen rugalmasan lehet alkalmazni a különböző adatelemzési feladatokhoz. Az alkalmazások tipikus területe azon ágak, ahol nagymennyiségű, összetett kapcsolatban álló adatok kerülnek letárolásra, viszonylag egységes, tiszta adatrendszerrel rendelkeznek, s igen jól hasznosíthatók ezen adatrendszerből levont következtetések. Így az OLAP, DW rendszerek elsősorban azon kiterjedt vevői, fogyasztói körrel rendelkező vállalatoknál kerülnek hasznosításra, ahol az egyes vevőket egyenként is nyilvántartásba veszik a hosszú távú kapcsolat céljából. Természetesen emellett számos más fontos alkalmazási terület található még, melyekről a következő listában adunk egy rövid áttekintést, megadva a megoldandó feladat jellegét és azon területeket ahol ez a feladat tipikusan előfordul:

- Piac elemzési feladatok: Ez a legelterjedtebb alkalmazási terület az OALP rendszerek számára. Ezen feladaton belül a fogyasztók, vevők szokásainak a vizsgálata, a trendek felismerése, a piac várható alakulásának felderítése szokott

konkrét tevékenységként megjelenni. A piac elemzési tevékenységek igen nagy jelentőségű a különböző fogyasztási cikkeket készítő vállalatoknál, hiszen a termék palettájuk rendszerint igen széles, s gyorsan változik. A konkurens cégekkel való állandó piaci verseny közepette igen fontos hogy fel tudjanak készülni a várható piaci változásokra, s minél jobban megismerjék a vevők fogyasztási szokásait. Ezen a területen igen gyakran kell ellenőrizni a megvalósult folyamatokat kiigazítva az értékesítés stratégiai és taktikai elemeit. A gyártók mellett a kereskedő cégek is hasonló feladat előtt állnak, csak ők közvetlenül is kapcsolatba kerülnek a végfelhasználókkal. A piac elemzési munkát nagyban segíti az a tény, hogy napjainkban a hagyományos felméréseken alapuló piackutatási módszerek helyett a bankkártyák használatával egy újfajta információ gyűjtési lehetőség is megnyílott a vevők paramétereinek a minél alaposabb megismerése felé. A banki szolgáltatások, biztosító társaságok esetében is szükség van ilyen jellegű kiterjedt és intenzív piackutatási elemzésekre, s itt még az az előny is megvan, hogy ők sokkal részletesebb információkkal rendelkeznek az ügyfelekről. Mivel ezek a társaságok sokkal szorosabb kapcsolatban állnak a vevőikkel, mint a korábban említett vállalatok, ezért itt sokkal mélyebb, egyénekre jobban lebontott elemzési feladatok elvégzésére is szükség van.

- WEB-es felhasználói magatartás elemzés. A WEB-es felületű ügyfél - szolgáltató interface megjelenése a letárolható, összegyűjthető adatok mennyiségének robbanásszerű növekedését hozta magával. Ugyanis egyrészt megnőtt a kapcsolatba kerülő ügyfelek száma, hiszen az Internet révén szinte a világ bármely részéből pillanatok alatt bejelentkezhet a több száz millió Internet hozzáféréssel rendelkező potenciális vásárló, ügyfél a szolgáltató vagy kereskedő cég WEB lapjára. Mivel a kapcsolat teljes egészében programon keresztül történik, lehetőség van arra, hogy a számítógép feljegyezze a bejelentkezett ügyfél minden egyes elemi lépését, tevékenységét. Ezáltal sokkal több ügyfélről sokkal több és részletesebb adat áll rendelkezésre. A megnövekedett adatmennyiség manuális feldolgozása, elemzése reménytelen feladat, ezért ezen a területen a legésszerűbb megoldás az OLAP – DW által nyújtott szolgáltatások kiaknázása. Az adatok elemzése révén lehetőség nyílik az ügyfelek viselkedési szokásainak az felderítésére. A viselkedési szabályok alapján hatékonyabban átalakítható a vállalat WEB-es szolgáltatási köre és megjelenési formája. Az elvégzett módosítások hatása is pontosabban, több oldalról lemérhető az OLAP eszközök segítségével. Technikai oldalról is természetes fejlődést jelent a különböző újszerű informatikai technológiák ötvözése.
- A vevő menedzsment jellegű alkalmazások esetében kimondottan a vevők vásárlási szokásainak jobb megismerése, a vásárlói szokásokhoz igazodó piacpolitika kialakítása a cél. Ennek során fel kell tárni, hogy mely paraméterek szignifikánsak a vevői magatartás kialakulásában, milyen tulajdonságok jellemzik az egyes vevői csoportokat. Ezt követően kidolgozásra kerül, hogy milyen kedvezményekkel, ajánlatokkal célszerű szembesíteni a potenciális vevőkört, hogy annak minél nagyobb legyen a hatásfoka. Célszerű annak felderítése is, hogy az egyes törzsfogyasztók esetében mely árufajták, szolgáltatásfajták dominálnak, s ezekhez mely más áruk vagy szolgáltatások kapcsolhatók a forgalom növelése érdekében. Az elemzés során fel kell tárni, hogy mely vevői csoportokat lehet még bevonni a jövőben a törzsvásárlói körbe. Fontos annak felismerése, hogy a fogyasztók mely tulajdonságai játszanak meghatározó szerepet a vásárlásokban, s ez alapján lehet szelektálni azon fogyasztói kört, akiket érdemes direkt eszközökkel is megszólítani.

- Pénzügyi tervek készítése. Kicsit szokatlannak tűnik ez az alkalmazási terület, de itt is jól hasznosíthatók az OLAP által nyújtott adatelemzési segédeszközök. Ugyanis a DW rendszerben rendelkezésre állnak mindazon adatok, melyek a vállalat működését leírják, s ezekből az OLAP elemzési funkciói segítségével a különböző működési alternatívák modellezhetők, a működési adatok kiszámolhatók, s azok pénzügyi vonzatai meghatározhatók. A különböző alternatívák gyorsan kidolgozhatók, és több szinten is megvitathatók viszonylag rövid időn belül. Az elfogadható változat kiválasztása így sokkal gyorsabban lezajlik, mintha csak a hagyományos tervezési eszközökre támaszkodnánk.
- Pénzügyi jelentések elkészítése. A számviteli, pénzügyi szakemberek az első úttörő OLAP felhasználók közé tartoznak, hiszen nagy mennyiségű pénzügyi adattal kell dolgozniuk, melyekből számos, több különböző pénzügyi vetületet tartalmazó jelentést kell készíteniük. A multi-dimenzionalitás egy alapvető tulajdonsága ezen rendszereknek, hiszen egy jelentés során a pénzügyi adatok egységek, kategóriák, időszakok bontásban jelennek meg összevetve a tényleges és terv adatokat.
- Vezetői jelentések készítése: A vállalati vezetőknek szóló jelentések elkészítése során is ki lehet használni az OLAP nyújtotta lehetőségeket. A vezetői jelentésekben rendszerint az elemző adatok dominálnak a részletező adatok helyett, s egy globális, aggregált kép kialakulását támogatják. A jelentéseknek tehát átfogónak kell lenniük, a tendenciák feltárását kell támogatniuk, s alkalmasnak kell lenniük az eltérések kimutatására is. A jelentések rendszerint számos működési paraméter kiszámítását tartalmazzák, s számos összehasonlítás is a részük a tényleges adatoknak a tervezett adatokkal való összevetésére vonatkozólag.
- Nyereségesség elemzés. A vállalatok sikeres működésének egyik legfőbb mérőszáma a nyereségesség. Emiatt igen fontos a vállalatok számára hogy pontosan tudják honnan származik a nyereségességük vagy hogy honnan nem származott nyereségesség. A nyereséges tevékenységek meghatározása rendszerint nem is egyszerű feladat, hiszen a termelés közvetlen költségi, mint anyagdíj és közvetlen munkadíj egyre csökkenő részt kap a teljes költségen belül, miközben a közvetett költségek, mint a kutatás, marketing egyre nagyobb szereppel rendelkezik. A nehézséget még az is fokozza, hogy ezek a költségek nem csak egy termékhez kötődnek, ezért összetett módon szét kell osztani a különböző termékek között. A bonyolult kapcsolatokon alapuló számítások elvégzésében adnak segítséget a különböző OLAP elemzési funkciók.
- Minőségbiztosítás. A termelés, a szolgáltatások áruk minőségének folyamatos szinten tartása a vállalat egyik fontos tevékenysége. A termelésből, vevőszolgálattól kapott adatokra építve az OLAP elemzési funkciók segítségével meghatározhatók, hogy a többdimenziós leíró adatok változása alapján hol jelennek meg a minőség romlását okozó tényezők a rendszerben.

Az általános alkalmazási terület áttekintés után egy kicsit közelebbről is megnézzük az OLAP és DW rendszerek lehetőségeit. Ehhez áttekintjük egy konkrét OLAP rendszer paramétereit és moduljait A vizsgált rendszert az elterjedtségük alapján közismert rendszerek közül véletlenszerűen emeltünk ki. A modulok bemutatásával pontosabb képet kaphatunk, hogy milyen jellegű adatelemzési, adatkezelési célra is használhatók a piacon szereplő OLAP rendszerek. A bemutatáshoz a Sagent cég OLAP rendszerét vesszük példaként.

A Sagent OLAP rendszere széles körben átfogja az OLAP – DW alkalmazásokhoz szükséges tevékenységi, funkcionalitási köröket. A Sagent Solution csomag tartalmazza mindazon eszközöket melyek segítségével felépíthető, karbantartható egy DW rendszer, kezdve az adatok betöltésétől, tisztításától egészen az adatelemezések eredményének felhasználóbarát megjelenítéséig. A Sagent rendszer fő célja egy hatékony eszköz biztosítása a döntéstámogatási feladatok ellátására mind a vevő menedzsment mind a pénzügyi területek vállalatai számára. Mivel a Sagent csomag együtt tartalmazza az összes szükséges komponenst, a rendszer karbantartása is sokkal egyszerűbbé válik, hiszen nem kell törődni a különböző más helyről származó komponensek illesztésével, összehangolásával. A Sagent Portal csomag az OLAP – DW rendszerhez biztosít egy hatékony, testre szabott elérési felületet. Minden felhasználóhoz ki lehet alakítani az egyedi igényeinek megfelelő információs és funkcionális tartalmat a kapcsolat eszközéül használt WEB lapon. A Centrus Real Time csomag segítségével az ügyfelek, vevők elérési adatai, címei és telefonszámai, valamint a különböző geográfiai, statisztikai adatai tarthatók nyilván egy könnyen kezelhető, nyílt formában.

A Sagent Portal rendszer a következő fontosabb modulokat tartalmazza:

- Sagent Data Load Server
  Egy nagy teljesítményű szerver az adatoknak az adattárházba való betöltésére. A modul alkalmas a betöltés során fellépő adat kiemelési, transzformációs, adatbeviteli funkciók ellátására. A rendszer el tudja végezni a heterogén forrás adat környezetben történő adatbetöltési folyamatokat is.
- Sagent Data Access Server
  A modul lehetőséget ad arra, hogy a DW-ben tárolt adatokra vonatkozólag gyorsan és egyszerűen illeszteni lehessen a különböző multi dimenizonális felhasználói nézeteket.
- *i*Studio
  Az iStudio segítségével gyorsan generálhatók és lefuttathatók a DW rendszerre vonatkozó felhasználói lekérdezések és jelentések. Az *i*Studio modul egy felhasználóbarát WEB-es kezelő felülettel rendelkezik.
- Sagent Admin
  Egy hatékony, széles körű funkcionalitással rendelkező irányító, vezérlő modul, mely segítségével a rendszer adminisztrátorai GUI felületen keresztül felügyelni tudják a Sagent rendszer működését, paraméterezését.
- Sagent Automation
  Ez egy esemény vezérelt ütemező rendszer, mely segítségével automatizálhatók egyes rendszeres tevékenységek végrehajtása, s a rendszer teljesítmény, leterhelés paramétereinek figyelése.
- Sagent Design Studio
  Egy hatékony, grafikus fejlesztő környezet az OLAP-DW rendszerben lezajló adatáramlási folyamatok megtervezésére. Az érintett adatáramlás lefedi a teljes DW területet, kezdve az adatbetöltéstől, transzformációtól egészen a lekérdezésekig, jelentések generálásáig.
- Sagent Information Studio
  Ez a modul egy grafikus kliens oldali felület, mellyel a felhasználók gyorsan és egyszerűen tudnak lekérdezéseket összeállítani, az eredmények megjeleníteni, letárolni, s egy későbbi időpontban esetleg újra felhasználni.
- SAS® Solution for Enterprise Marketing Automation
  Ez a modul a vállalat piac és vevő menedzsment munkájának megsegítésére szolgál. A

rendszer alkalmazásával kijelölhetők a leghűségesebb vevők csoportja, célirányosabbá tehetők a piacfejlesztési lépések.

- SAS® Enterprise Miner™
  Az Enetrprise Miner egy intelligens módszereket alkalmazó adatbányászási modul, mely segítségével az összegyűjtött vevői, forgalmi adatok alapján feltárhatók a forgalmi adatokban rejlő szabályszerűségek. A rendszer segítségével kimutathatók az adatok változásában rejlő tendenciák, s meghatározhatók a rendellenes viselkedést mutató folyamatok
- Sagent Forecaster Transform
  Ezen előrejelzési modul alkalmazásával a DW-ben tárolt történeti adatok alapján meghatározható az egyes működési paraméterek változási tendenciája, s ezáltal megjósolhatók a jövőben valószínűsíthető folyamatok is.
- Sagent Address Cleanser & Coder Transform
  Az adattárházba bejövő cím és földrajzi adatok tisztítására szolgál ez a transzformációs modul. Segítségével elérhető, hogy egységes formában, helyes tartalommal kerülnek majd letárolásra az egyes elérési, földrajzi adatok, mely egyik alapfeltétele a pontos információ szolgáltatásnak és rekordillesztési feladatoknak.
- DirectLink™ for R/3
  A modul szerepe hogy gyorsan és egyszerűen meg tudjuk oldani az adattárházunk feltöltését az SAP-R3 információs rendszer adatiból átvett adatokkal.
- Sagent Analysis
  Ez a modul egy hatékony és könnyen kezelhető adatelemzési komponens, melyet alkalmazva az egyes adatkockákon elvégezhetők a megismert adatkocka műveletek , így a szeletelések, szűrések, drill down és drill up műveletek, vagy éppen a keresztreferencia táblák létrehozatala.
- Sagent Reports
  Hatékony és könnyen kezelhető eszköz a nyomtatási és képernyő jelentések megtervezésére és futtatására.
- Sagent WebLink
  A modul egy WEB alapú kapcsolati felületet biztosít a rendszerben tárolt adatokhoz, biztosítva az egyszerű lekérdezéstől kezdve a bonyolultabb adatelemzési feladatokig a szokásos OLAP műveleteket.
- Analytical Calculator
  Ezen komponens számos statisztikai segédfüggvényt, számítási módszert biztosít a felhasználók részére az összetettebb statisztikai jellegű elemzések elvégzésére.

A felsorolt komponensek mellett bizonyos más termékekben még külön modulként fordul elő

- az alkalmazás tervező modul, melyben OLAP feladatokat ellátó alkalmazói programok készíthetők gyors és hatékony módon, valamint egy
- védelmi rendszer felügyelő modul, mellyel az OLAP – DW rendszer hozzáférési jogosultság rendszere alakítható ki

A fenti áttekintés alapján már pontosabban láthatjuk, hogy milyen szolgáltatásokat remélhetünk egy piaci terméktől. Természetesen ügyelni kell arra, hogy a különböző piaci termékek igen lényegesen különbözhetnek egymástól a nyújtott szolgáltatások, OALP, DW funkciók tekintetében. A piacon sok olyan OLAP rendszer is található, mely nem a multidimenzionális adatmodellre épült, hanem megmaradt a hagyományos relációs modell keretein belül. Az OLAP – DW rendszer kiválasztásakor figyelembe kell venni a

- a működési platformot

- a nyújtott szolgáltatásokat
- az adatkapcsolatok szabványosságát
- a nyitott adatkapcsolati és fejlesztési lehetőségeket
- a rendszer rugalmasságát, paraméterezési lehetőségét
- felhasználóbarát kezelő felületét
- a cég által biztosított karbantartást
- a termék elterjedtségét

A választáshoz több cég ajánlatát is célszerű áttekinteni. Ezen ajánlatok begyűjtését megkönnyítendő, az információk hatékonyabb és gyorsabb megszerzését támogatandó a következő részben megadjuk a fontosabb piaci termékek WEB-es elérési címeit és emellett felsorolunk néhány olyan WEB címet is, ahol további lényeges információk gyűjthetők össze az OLAP rendszerekkel kapcsolatban.

## Áttekintés a piaci termékekről

Elsőként néhány ismertebb OLAP – DW rendszer adatait adjuk meg a WEB-es elérési címmel, ahonnan további bővebb információk szerezhetők be.

| Cég | web cím | terület |
|---|---|---|
| Access | www.microsoft.com | jelentés és lekérdezés generáló |
| Active Reports | www.datadynamics.com | jelentés és lekérdezés generáló |
| Approach | www.software.ibm.com | jelentés és lekérdezés generáló |
| Brio O NE | www.brio.com | jelentés és lekérdezés generáló és MD DW |
| Business Objects | www.businessobjects.com | relációs DW |
| CA Easytrieve | www.cai.com | jelentés és lekérdezés generáló |
| Crystal Analysis | www.crystaldecisions.com | MD és relációs DW |
| Crystal Reports | www.crystaldecisions.com | jelentés és lekérdezés generáló |
| Cubeware | www.cubeware.de | MD és relációs DW |
| Data Cruiser | www.specialware.com | jelentés és lekérdezés generáló |
| Decisonbase OLAP | www.cai.com | MD és relációs DW |
| DB2 OLAP server | www.software.ibm.com | relációs DW |
| Elixir Report | www.elixirtech.com | jelentés és lekérdezés generáló |
| Decisonbase Reporter | www.platinium.com | jelentés és lekérdezés generáló |
| Enterprise-Integrator | www.apertus.com | adat tisztítás fuzzy logikára építve |
| Essbase OLAP | www.hyperion.com | MD DW |
| ETI Extract | www.evtech.com | adat betöltés |
| Focus Six | www.ibi.com | jelentés és lekérdezés generáló |
| Fusion | www.ibi.com | MD DW |
| Fuzzyquery | www.sonalysts.com | jelentés és lekérdezés generáló |
| HotQuery | www.hotquery.com | jelentés és lekérdezés generáló |
| Impromptu | www.cognos.com | jelentés és lekérdezés generáló |
| InfoMaker | www.sybase.com | jelentés és lekérdezés generáló |

| | | |
|---|---|---|
| Innovative | www.innovgrp.com | adat tisztítás |
| Integrity | www.vality.com | adat tisztítás |
| MS Analysis Services | www.microsoft.com | MD és relációs DW |
| MetaCube | www.informix.com | relációs DW |
| MicroStrategy | www.microstrategy.com | relációs DW |
| Object Reports | www.rollinssoft.com | jelentés és lekérdezés generáló |
| Oracle Discoverer | www.oracle.com | jelentés és lekérdezés generáló |
| Oracle Advanced Analytic | www.oracle.com | MD és relációs DW |
| Paradox | www.corel.com | jelentés és lekérdezés generáló |
| Passport | www.carleton.com | adatbetöltés különböző forrásokból, adattisztítási eszközök |
| PowerOlap | www.rolap.com | MD DW |
| PowerPayl | www.cognos.com | MD és relációs DW |
| QDB | www.dqb.com | adat tisztítás |
| QMF | www.software.ibm.com | jelentés és lekérdezés generáló |
| ReportTool | www.zensoftware.com | jelentés és lekérdezés generáló |
| Sagent | www.sagenttech.com | jelentés és lekérdezés generáló és DW |
| SAP | www.sap.com | DW |
| SAS Enterprise Reporter | www.sas.com | jelentés és lekérdezés generáló és DW |
| Seagate Info | www.crystaldecisions.com | MD DW |
| Trillium | www.trilliumsoft.com | adat tisztítás, adatbányászás |
| Warehouse Manager | www.prismasolution.com | adat áttöltés, betöltés és transzformáció |
| Vision | www.vision.sterling.com | jelentés és lekérdezés generáló |
| Visual FoxPro | www.microsoft.com | jelentés és lekérdezés generáló |
| WizSoft | www.wizsoft.com | adat tisztítás, feltárja a szabályokat és az azoktól való eltéréseket |

A gyártók adatai mellett néhány hasznos az OLAP és DW területhez kapcsolódó WEB címet is javaslunk további információ szerzés céljából felkeresni.

Data Warehousing Information Center www.dwinfocenter.org

egy igen hasznos forrás, igen sok különböző hivatkozással a kapcsolódó területekre. Ebből a lapból kiindulva mintegy 80 kapcsolatot bejárva teljesen lefedhetjük e tématerület legfontosabb Web-es információforrásait. Minden fontosabb tématerületről egy általános ismertetőt is tartalmaz ez a WEB hely. Az érintett tématerületek listája:

**Getting Started with Learning About Data Warehousing**
**A Definition of Data Warehousing**
**A Definition of Decision Support**
**The Case for Data Warehousing**

**The Case Against Data Warehousing**
**Actions for Data Warehouse Success**
**Data Warehousing Gotchas**
**Performing Data Warehousing Software Evaluations**
**An (Informal) Taxonomy of Data Warehouse Data Errors**
**Data Warehousing Political Issues**
**Different Aspects of Data Warehouse Architecture**
**What to Learn About in Order to Speed Up Data Warehouse Querying**
**What to Learn About in Order to Speed Up Data Warehouse Loading**
**How to Save Money on Your Data Warehousing Efforts**
**Using Data Warehousing in Strategic Decision Making**
**Maintenance Issues for Data Warehousing Systems**
**What Decision Support Tools are Used For**
**Is Web Data Analysis (i.e., Web Data Mining) Different?**

A lapon az elméleti anyag mellett az egyes cégek, termékek WEB címei is elérhetők.

OLAP központ www.olapcouncil.org

Az OLAP központ 1995-ben alakult az ipar és a érdeklődő magánszemélyek OLAP alkalmazásokkal kapcsolatos információs igényeinek kiszolgálására. Ez a lap is számtalan oktató anyagot és termék összehasonlító, ismertetető leírást tartalmaz.