Példák a Microsoft SqlServer szövegkezelő moduljához

1. példa (ld. 262. old. alja)

Az alábbi példákkal az SQLSERVER keresési lehetőségét mutatjuk be. Kifejezéskeresésnél kettős idézőjelek közé kell tenni a keresőkifejezést:

```
SELECT szoveg FROM doksik WHERE CONTAINS(szoveg, ' "New York" ')
```

A prefix kifejezésnél a szavak elejét rögzítjük, a végződésük tetszőleges lehet. A mintát szintén kettős idézőjelben kell megadni, és a minta végén szerepelnie kell a * dzsókerkarakternek. Így például a

```
"tenge*"
```

minta illeszkedik a tenger és tengerpart szavakra is.

A származtatási kifejezésnél egy megadott alapszó származtatott alakjait is elfogadja az illesztő motor. A kifejezés alakja:

```
FORMSOF (mód, elemi kifejezés)
```

Kétféle származtási mód támogatott:

- INFLECTIONAL: a megadott elemi kifejezést tekinti szótőnek,
- THESAURUS: a megadott elemmel szinonimakifejezések elfogadása.

Ez a keresési változat csak a támogatott nyelvek esetében használható.

A tezauruszállományok XML-formátumban adják meg a szavakat és szinonimáikat. A tezaurusz felépítését az alábbi XML-séma írja le:

```
<thesaurus xmlns="x-schema:tsSchema.xml">
<expansion>
<sub> kifejezés1 </sub>
<sub> kifejezés2 </sub>
...
</expansion>
<replacement>
<pat> kifejezés3 </pat>
<sub> kifejezés4 </sub>
<sub> kifejezés5 </sub>
...
</replacement>
</thesaurus>
```

A leírásban kétféle szekció szerepel: a kiterjesztést és a helyettesítést megadó rész. A kiterjesztési szekció az <expansion> elemmel határolt. Az itt szereplő elemek egymás szinonimáinak tekinthetők. A szinonim szavak, kifejezések a

<sub> elemmel határoltak. Ha az illesztési feltételben ezen elemek egyike előfordul, akkor a keresőmotor minden olyan kifejezést illeszkedőnek tekint, ahol a szinonim kifejezések egyike előfordul. A helyettesítő szekciót a <replacement> elem határolja. Itt egy mintakifejezés, és egy vagy több helyettesítő kifejezés adható meg. A mintakifejezést a <pat> elem határolja. Ekkor a keresési feltételben szereplő mintakifejezést helyettesíti a <sub> elemekben megadott helyettesítő kifejezésekkel, és a keresés ezekre fog lefutni.

Példaként vegyük az alábbi tezauruszállományt:

```
<thesaurus xmlns="x-schema:tsSchema.xml">
<expansion>
<sub> kutya </sub>
<sub> eb </sub>
</expansion>
<replacement>
<pat> csahos </pat>
<sub> kutya </sub>
<sub> véreb </sub>
</replacement>
</thesaurus>
```

Az első rész azt jelöli, hogy a kutya és az eb kifejezések ekvivalensek egymással. Ha a keresésben a

```
CONTAINS (leiras, ' eb ')
```

alakot használjuk, akkor pontos egyezés esetén csak az eb szó lesz illeszkedő. Ha azonban a közelítő keresésre szolgáló

```
CONTAINS(leiras,' FORMSOF(THESAURUS, eb) ')
```

operátort használjuk, akkor az eb mellett a kutya is megfelelő lesz, mivel a tezaurusz alapján ez szinonimája az eb szónak. Ha a kérdést

```
CONTAINS(leiras,' FORMSOF(THESAURUS, csahos) ')
```

alakban fogalmazzuk meg, akkor a csahos szó nem kerül be az eredménybe, csak az eb, kutya és véreb alakok jöhetnek számításba.

A közelség alapú kifejezéseknél a megadott kifejezéseket úgy kell tartalmaznia a szövegnek, hogy az egyes kifejezések a pozíció alapján közel helyezkedjenek el egymáshoz. Ekkor a keresőkifejezést a

```
kifejezés_1 NEAR kifejezés_2
```

alakban adjuk meg. A következő lekérdezésben azon dokumentumokat keressük, melyekben a football és Hungary szavak egymáshoz közel fordulnak elő:

```
SELECT * FROM doksik WHERE CONTAINS (szoveg, 'football NEAR Hunga-
ry')
```

A súlyozott kifejezéseknél megadhatjuk, hogy egyszerre több keresőkifejezés esetén milyen fontosságot rendelünk az egyes kifejezésekhez. A kulcsszavak mellett azok súlyértékét is definiálhatjuk az illesztési operátornál. Az illesztéshez ebben az esetben az ISABOUT operátort kell használni:

ISABOUT (*kif*$_1$ WEIGHT(*érték*$_1$), *kif*$_2$ WEIGHT(*érték*$_2$), . . .)

ahol a súlyérték 0 és 1 közötti valós szám lehet. Az eredő súlyérték az eredményhalmaz sorrendjét befolyásolja, tartalmára nincs hatással. A rangsor akkor jelenik meg, amikor az eredménylistát a CONTAINSTABLE függvénnyel kérjük le.

Elemi kifejezésekből összetett kifejezést képezhetünk az

- AND: és;
- AND NOT: és nem;
- OR: vagy

operátorokkal. Így például a

```
CONTAINS (mezo, ' idared OR FORMSOF(INFLECTIONAL,alma) ')
```

esetében az illesztés feltétele az, hogy vagy az idared szó legyen a szövegben, vagy egy olyan kifejezés szerepeljen, melynek szótöve az alma szó.

2. példa (ld. 263. old. közepe)

Ha például az irat táblában a leiras mező a szöveges adatokat, és a kod mező az egyedi azonosítót tárolja, akkor az indexet az alábbi módon hozhatjuk létre:

```
CREATE UNIQUE INDEX irati1 ON irat (kod);
```

A Fulltext-index tényleges létrehozása az alábbi paranccsal lehetséges:

```
CREATE FULLTEXT INDEX ON irat (leiras)
key INDEX irati1 ON indexkatalógus;
```

A létrehozott index alapján lekérdezhetjük az első 10 iratot, melynél a leiras mezőben szerepel az adatbázis szóval szinonim kifejezés.

```
SELECT irat.leiras, seged.rank
FROM irat INNER JOIN
CONTAINSTABLE (irat, leiras,
' FORMSOF(THESAURUS,adatbázis)', 10) AS seged
ON irat.kod = seged.key
ORDER BY seged.rank DESC;
```

A CONTAINSTABLE függvényt a FROM részben adtuk meg és az eredményhez a seged alias értéket rendeltük hozzá. A seged tábla key mezőjén keresztül azonosíthatjuk a kapcsolódó alaptáblarekordot. A rank mező a rekord rangsorbeli pozícióját mutatja. Az eredménytábla rank mezőjében szereplő súlyértéknél a nagyobb számérték nagyobb relevanciaértéket jelöl.

3. példa (ld. 263. old. alja)

```
SELECT irat.leiras, seged.rank
FROM irat INNER JOIN
FREETEXTTABLE (irat, leiras,'adatbázis', 10) AS seged
ON irat.kod = seged.key
ORDER BY seged.rank DESC;
```

Ez a lekérdezés azokat a rekordokat adja vissza, ahol az adatbázis szónak valamely rokon alakja fordul elő a leiras mezőben. A sorrendképzés és limitálás formátuma és jelentése megegyezik a CONTAINSTABLE megfelelő elemével.