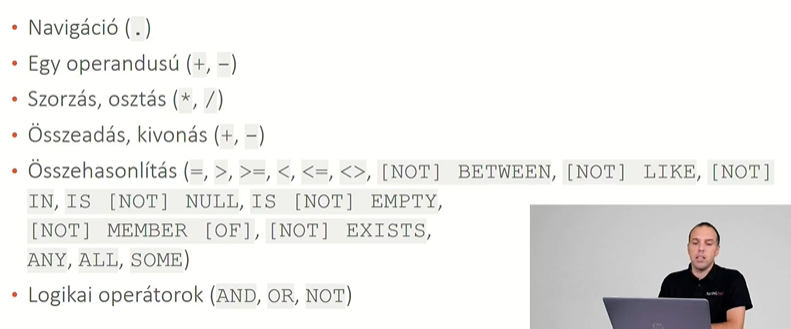
**Haladó lekérdezések**

**Milyen operátorok használhatóak a lekérdezésekben?**



*A lekérdezések feltétel részében a következő operátorokat használhatjuk:*

Navigáció pont (.) operátorral. A navigáció itt az attribútumok közötti navigációt jelenti egy path expression (bejárási útvonalat tartalmazó kifejezés) esetén.

Egy operandusú operátorok (+ és -).

Két operandusú operátorok az összeadás, kivonás, szorzás és osztás jelei (+, -, \* és /).

Összehasonlító operátorok. Egyenlőséget és relációkat vizsgáló operátorok (=, <, <=, >, >=, <>).

A [NOT] BETWEEN két érték közötti elhelyezkedést vizsgál.

A [NOT] LIKE két karakterlánc hasonlóságát vizsgálja, itt használhatóak wildcard karakterek (például %).

Az IN-nel azt lehet vizsgálni, hogy egy érték szerepel-e egy Collection típusú változó értékei között.

A null értékek vizsgálatára az IS [NOT] NULL operátor használható.

Egy Collection típusú változó ürességét nézhetjük meg az IS [NOT] EMPTY operátorral.

A [NOT] MEMBER OF-fal azt lehet megnézni, hogy egy elem egy másik Collection-ben szerepel-e.

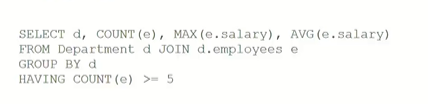
Egy allekérdezés esetén annak visszatérési értékét (tehát hogy hoz-e vissza rekordot) vizsgálhatjuk a [NOT] EXISTS operátorral.

Az allekérdezésekre feltételt is adhatunk meg, majd megnézhetjük, hogy az allekérdezés által visszaadott eredményhalmazban van-e egyáltalán olyan elem, amely megfelel a feltételnek (ANY), vagy mindegyik megfelel-e (ALL), esetleg csak néhány (SOME, számok esetén).

Logikai operátorok (AND, OR, NOT).

**Aggregált lekérdezések**

**(aggregáló fügvények használatakor(cout, max, avg); having= where, ha előtte group by volt)**



Írhatunk aggregált lekérdezéseket az aggregáló függvények használatával:

A rekordok összeszámolása a COUNT függvénnyel történhet.

A legnagyobb vagy legkisebb érték megkeresése a MAX és a MIN függvények segítségével.

Az AVG az átlagos értéket számolja ki.

Lehetőség van az értékek csoportosítására a GROUP BY kulcsszavak használatával.

A különböző csoportokra feltételeket is adhatunk meg a HAVING kulcsszó használatával.

Az alábbi példában látható lekérdezés visszaadja department-enként az ott dolgozó alkalmazottak számát, maximális fizetését és átlagos fizetését:

SELECT d, COUNT(e), MAX(e.salary), AVG(e.salary)

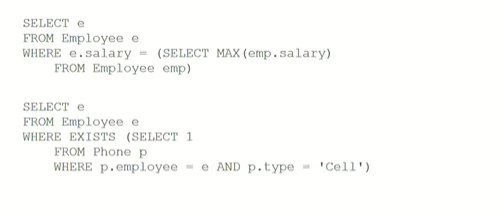
FROM Department d JOIN d.employees e

GROUP BY d

HAVING COUNT(e) >= 5

**Allekérdezések**

**(=selectben select)**



*Allekérdezések használatára álljon itt a következő két példa:*

SELECT e

FROM Employee e

WHERE e.salary = (SELECT MAX(emp.salary)

FROM Employee emp)

Ez visszaadja azt az Employee-t, amelyiknek a legnagyobb a fizetése (salary). A **belső SELECT lekérdezi a legmagasabb fizetést, a külső pedig azt az alkalmazottat,** akinek ezzel egyezik meg a fizetése.

SELECT e

FROM Employee e

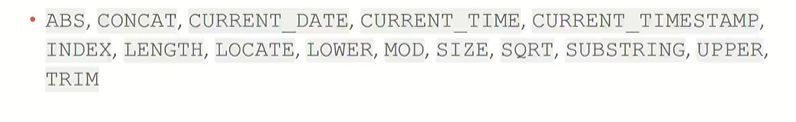
**WHERE EXISTS** (SELECT 1

FROM Phone p

WHERE p.employee = e AND p.type = 'Cell')

Látható, hogy itt a **külső SELECT-ben található identification variable-t (e) használja a belső SELECT is.** A belső lekérdezésben feltétel található arra vonatkozóan, hogy az alkalmazottnak legyen megadva a telefonszáma és az mobiltelefon típusú legyen (p.type = 'Cell'), a külső lekérdezés pedig ezekre az alkalmazottakra kérdez rá. Vagyis rövidebben: ez a lekérdezés lekérdezi azokat az alkalmazottakat, amelyeknek mobiltelefonja van.

**Beépített függvények**

****

A JPQL-ben megtalálható jó pár beépített függvény is:

ABS: abszolútérték-függvény.

CONCAT: ezzel Stringeket tudunk konkatenálni.

CURRENT\_DATE, CURRENT\_TIME, CURRENT\_TIMESTAMP: az aktuális dátumot, időt és timestampet adja vissza, ezek az adatbázis időadataival dolgoznak.

INDEX: megmondja egy adott elemről, hogy a Collection hányadik pozícióján áll (vagyis mi az indexe). Ez természetesen csak sorrendezett kollekcióknál működik.

LENGTH: egy Stringnek a hosszát adja vissza.

LOCATE: meg lehet nézni, hogy egy adott String szerepel-e substringként egy másik Stringben, illetve hányadik pozíción.

LOWER: kisbetűsít egy Stringet.

MOD: maradékos osztás.

SIZE: az adott kollekció méretét, elemszámát adja vissza.

SQRT: négyzetgyökvonás.

SUBSTRING: egy String egy részét lehet lekérdezni.

UPPER: nagybetűsít egy Stringet.

TRIM: a String két oldaláról levágja a whitespace karaktereket.

A JPA ezeket a függvényeket természetesen átfordítja a konkrét adatbáziskezelő megfelelő függvényeire.

**Order By**

Az alábbi példában az látható, hogy hogyan lehet rendezni a találatokat:

SELECT e, d

FROM Employee e JOIN e.department d

ORDER BY d.name, e.name DESC

Az SQL-hez hasonlóan az ORDER BY használatával lehet rendezést megadni. **Több mezőt is fel lehet egymás után sorolni vesszővel elválasztva, sőt, különböző entitásokban is szerepelhetnek ezek a felsorolt attribútum értékek.** Az alapértelmezett a növekvő rendezés, de a DESC kulcsszó használatával csökkenő rendezést is megadhatunk.

**CASE**

Ebben a példában pedig az látható, hogy a CASE kulcsszóval feltételt adhatunk meg, a különböző eseteket a WHEN kulcsszóval adhatjuk meg, és van egy ELSE ág is. A CASE-t az END kulcsszóval kell lezárni:

SELECT p.name,

CASE WHEN TYPE(p) = DesignProject THEN 'Development'

WHEN TYPE(p) = QualityProject THEN 'QA'

ELSE 'Non-Development'

END

FROM Project p

WHERE p.employees IS NOT EMPTY

Ez a lekérdezés a **TYPE függvényt is használja, amely megmondja egy adott változóról, hogy milyen típusú, azaz milyen osztályú entitás.**

**Kérdések**

*Milyen operátorok használhatóak a lekérdezésekben?*

*Hogy működikma navigáció operátor?*

*Milyen összehasonlító operátortokat ismersz?*

*Milyen logikai operátortokat ismersz?*

*Mik azok az aggregált lekérdezések?*

*Milyen aggregáló függvényeket ismersz?*

*Hogy csoportosítod az aggregált értékeket?*

*Hogy adhatsz meg aggregált lekérdezésnél csoportosítás után további feltételt?*

*Mi az az allekérdezés?*

*Mivel vizsgálhatjuk meg, hogy egy allekérdezés hoz-e vissza rekordot?*

*Milyen JPQL beépített függvényeket ismersz?*

*Hogyna lehet rendezni a találatokat?*

*Mire jó a CASE és hogy kell használni?*