Adatok szűrése, rendezése

Célkitűzések

- Szűrést kifejező lekérdezések végrehajtása
- A lekérdezés eredményének rendezése
- &változó használata *i*SQL*Plus-ban futási időben megadható feltételek céljából

A lista korlátozása kiválasztással

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90
103	Hunold	IT_PROG	60
104	Ernst	IT_PROG	60
107	Lorentz	IT_PROG	60
124	Mourgos	ST_MAN	50

...

20 rows selected.

Csak a 90-es osztály dolgozóit listázzuk ki!

EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90

Az eredmény korlátozása WHERE feltétellel

- A `WHERE` feltétel használata:

```
SELECT *|{[DISTINCT] column|expression [alias],...}
FROM   table
[WHERE condition(s)];
```

- A `WHERE` feltétel a `FROM` után következik.
- A feltétel oszlopértékeket, konstansokat, aritmetikai kifejezéseket, függvényértékeket hasonlíthat össze.
- Három részből áll:
  - Oszlopnév
  - Összehasonlítási feltétel
  - Oszlopnév, konstans vagy értékhalmaz

A WHERE feltétel használata

```
SELECT employee_id, last_name, job_id, department_id
FROM   employees
WHERE  department_id = 90 ;
```

EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90

Karakterláncok és dátumok

- A karakterláncokat és dátumokat egyszeres idézőjelek közé tesszük.
- A karakteres értékek kisbetű-nagybetű érzékenyek.
- A dátumértékek formátumérzékenyek.
- Az alap dátumformátum DD-MON-RR.

```
SELECT last_name, job_id, department_id
FROM   employees
WHERE  last_name = 'Whalen' ;
```

Összehasonlító feltételek

```
… WHERE kifejezés összehasonlítás érték
```

Összehasonlítási művelet	Jelentés
=	Egyenlő
>	Nagyobb mint
>=	Nagyobb vagy egyenlő mint
<	Kisebb mint
<=	Kisebb egyenlő mint
<> vagy != vagy ^=	Nem egyenlő
BETWEEN ...AND...	A két érték közé esik (zárt intervallum)
IN(halmaz)	Megyezik a halmaz valamelyik elemével
LIKE	Illeszkedik egy karakteres mintára
IS NULL	Az értéke nullérték

Összehasonlító feltétel használata

```
SELECT last_name, salary
FROM   employees
WHERE  salary <= 3000 ;
```

LAST_NAME	SALARY
Matos	2600
Vargas	2500

A BETWEEN feltétel használata

A BETWEEN feltétellel megadhatók azok a dolgozók, akiknek a fizetése 2500 és 3500 dollár közé esik:

```
SELECT last_name, salary
FROM   employees
WHERE  salary BETWEEN 2500 AND 3500 ;
```

Alsó korlát **Felső korlát**

LAST_NAME	SALARY
Rajs	3500
Davies	3100
Matos	2600
Vargas	2500

Az IN feltétel használata

Az IN tartalmazási feltétellel megadhatók azok a dolgozók, akiknek a főnöke 100, 101 vagy 201 azonosítóval rendelkezik:

```
SELECT employee_id, last_name, salary, manager_id
FROM   employees
WHERE  manager_id IN (100, 101, 201) ;
```

EMPLOYEE_ID	LAST_NAME	SALARY	MANAGER_ID
202	Fay	6000	201
200	Whalen	4400	101
205	Higgins	12000	101
101	Kochhar	17000	100
102	De Haan	17000	100
124	Mourgos	5800	100
149	Zlotkey	10500	100
201	Hartstein	13000	100

8 rows selected.

A LIKE feltétel használata

- A LIKE feltételben adott mintában a karaktereken kívül dzsókereket is lehet használni:
  - % nulla vagy több karaktert jelöl,
  - _ pontosan egy karaktert jelöl.

Milyen keresztnevű dolgozóknak kezdődik a keresztneve S-sel (nagy S-sel)?

```
SELECT   first_name
FROM     employees
WHERE    first_name LIKE 'S%';
```

A LIKE feltétel használata

- Több dzsókert is lehet a mintában használni.
- Milyen vezetéknevű dolgozók vezetéknevének 2. betűje „o”?

```
SELECT last_name
FROM   employees
WHERE  last_name LIKE '_o%' ;
```

LAST_NAME
Kochhar
Lorentz
Mourgos

- A % vagy _ dzsókerek ESCAPE karakterrel megjelölve közönséges karakterekké válnak.
- Kiknek az azonosítójuk kezdődik az SA_ három karakterrel?

```
SELECT employee_id, last_name, job_id
FROM   employees WHERE  job_id LIKE '%SA\_%' ESCAPE '\';
```

A nullértékek ellenőrzése

A nullértékekeket **IS NULL** vagy **IS NOT NULL** feltétellel tesztelhetjük.

```
SELECT last_name, manager_id
FROM   employees
WHERE  manager_id IS NULL;
```

LAST_NAME	MANAGER_ID
King	

A nullérték ismeretlen, meghatározatlan értéket jelöl, ezért nem egyenlő semmilyen értékkel, így a nullértéket nem lehet = vagy <> segítségével tesztelni!

Logikai feltételek

Művelet	Jelentés
AND	Igaz (TRUE), ha a feltétel mindkét tagja igaz
OR	Igaz (TRUE), ha a feltételnek legalább az egyik tagja igaz
NOT	Igaz (TRUE), ha a feltétel hamis

Az AND művelet használata

Mely dolgozók beosztásában szerepel a MAN karakterlánc és legalább 10000 a fizetésük?

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary >=10000
AND    job_id LIKE '%MAN%' ;
```

EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
149	Zlotkey	SA_MAN	10500
201	Hartstein	MK_MAN	13000

Az AND igazságtáblája nullértékkel kiegészítve:

AND	TRUE	FALSE	NULL
TRUE	TRUE	FALSE	NULL
FALSE	FALSE	FALSE	FALSE
NULL	NULL	FALSE	NULL

Az OR művelet

Mely dolgozók beosztásában szerepel a MAN karakterlánc vagy legalább 10000 a fizetésük?

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary >= 10000
OR     job_id LIKE '%MAN%' ;
```

EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
100	King	AD_PRES	24000
101	Kochhar	AD_VP	17000
102	De Haan	AD_VP	17000
124	Mourgos	ST_MAN	5800
149	Zlotkey	SA_MAN	10500
174	Abel	SA_REP	11000
201	Hartstein	MK_MAN	13000
205	Higgins	AC_MGR	12000

8 rows selected.

Az OR igazságtáblája nullértékkel kiegészítve:

OR	TRUE	FALSE	NULL
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	NULL
NULL	TRUE	NULL	NULL

A NOT művelet használata

Tetszőleges feltétel tagadható: NOT IN, NOT LIKE, NOT BETWEEN, IS NOT NULL.

Mely dolgozók beosztása se nem IT_PROG, se nem ST_CLERK, se nem SA_REP?

```
SELECT last_name, job_id
FROM   employees
WHERE  job_id
       NOT IN ('IT_PROG', 'ST_CLERK', 'SA_REP') ;
```

LAST_NAME	JOB_ID
King	AD_PRES
Kochhar	AD_VP
De Haan	AD_VP
Mourgos	ST_MAN
Zlotkey	SA_MAN
Whalen	AD_ASST
Hartstein	MK_MAN
Fay	MK_REP
Higgins	AC_MGR
Gietz	AC_ACCOUNT

10 rows selected.

A NOT igazságtáblája nullértékkel kiegészítve:

NOT	TRUE	FALSE	NULL
	FALSE	TRUE	NULL

A műveletek sorrendjének alapértelmezése

A kiértékelés sorrendje	A kifejezésben szereplő műveletek
1.	Aritmetikai műveletek
2.	Karakterlánck összefűzése (konkatenáció)
3.	Összehasonlító feltételek
4.	`IS [NOT] NULL`, `LIKE`, `[NOT] IN`
5.	`[NOT] BETWEEN`
6.	Nem egyenlő
7.	`NOT` logikai feltétel
8.	`AND` logikai feltétel
9.	`OR` logikai feltétel

A sorrendet zárójelezéssel felülírhatjuk!

Precedenciaszabályok alkalmazása

Kik azok, akik vagy 15000-nél többet kereső elnökök (AD_PRES), vagy mindegy mennyit keresnek, de a beosztásuk képviselő (SA_REP)? 1

```
SELECT last_name, job_id, salary
FROM   employees
WHERE  job_id = 'SA_REP'
OR     job_id = 'AD_PRES'
AND    salary > 15000;
```

1

LAST_NAME	JOB_ID	SALARY
King	AD_PRES	24000
Abel	SA_REP	11000
Taylor	SA_REP	8600
Grant	SA_REP	7000

```
SELECT last_name, job_id, salary
FROM   employees
WHERE  (job_id = 'SA_REP'
OR     job_id = 'AD_PRES')
AND    salary > 15000;
```

2

2

LAST_NAME	JOB_ID	SALARY
King	AD_PRES	24000

Kik azok, akik 15000-nél többet keresnek és vagy elnökök (AD_PRES) vagy képviselők (SA_REP)?

Az ORDER BY használata

Szintaxis: SELECT *kifejezés*
FROM *tábla*
[WHERE *feltétel(ek)*]
[ORDER BY {*oszlop, kifejezés, oszlopsorszám*} [ASC|DESC]];

- **Az eredmény sorai az ORDER BY szerint rendezettek:**
  - **ASC: növekvő sorrend, (alapértelmezés)**
  - **DESC: csökkenő sorrend**
- **Az ORDER BY rész a SELECT utasítás végén szerepel.**
- **Az ORDER BY nélkül a sorrend nem determinisztikus.**

```
SELECT   last_name, job_id, department_id, hire_date
FROM     employees
ORDER BY hire_date ;
```

LAST_NAME	JOB_ID	DEPARTMENT_ID	HIRE_DATE
King	AD_PRES	90	17-JUN-87
Whalen	AD_ASST	10	17-SEP-87
Kochhar	AD_VP	90	21-SEP-89
Hunold	IT_PROG	60	03-JAN-90
Ernst	IT_PROG	60	21-MAY-91

...

20 rows selected.

A belépési dátum szerint rendezett lista.

A rendezés használata

A nullértékek növekvő rendezés esetén a lista végén szerepelnek!

- **Csökkenő sorrend a belépés dátuma szerint:**

```
SELECT   last_name, job_id, department_id, hire_date
FROM     employees
ORDER BY hire_date DESC ;
```

1

- **Számított oszlop másodneve szerinti rendezés:**

```
SELECT employee_id, last_name, salary*12 annsal
FROM   employees
ORDER BY annsal ;
```

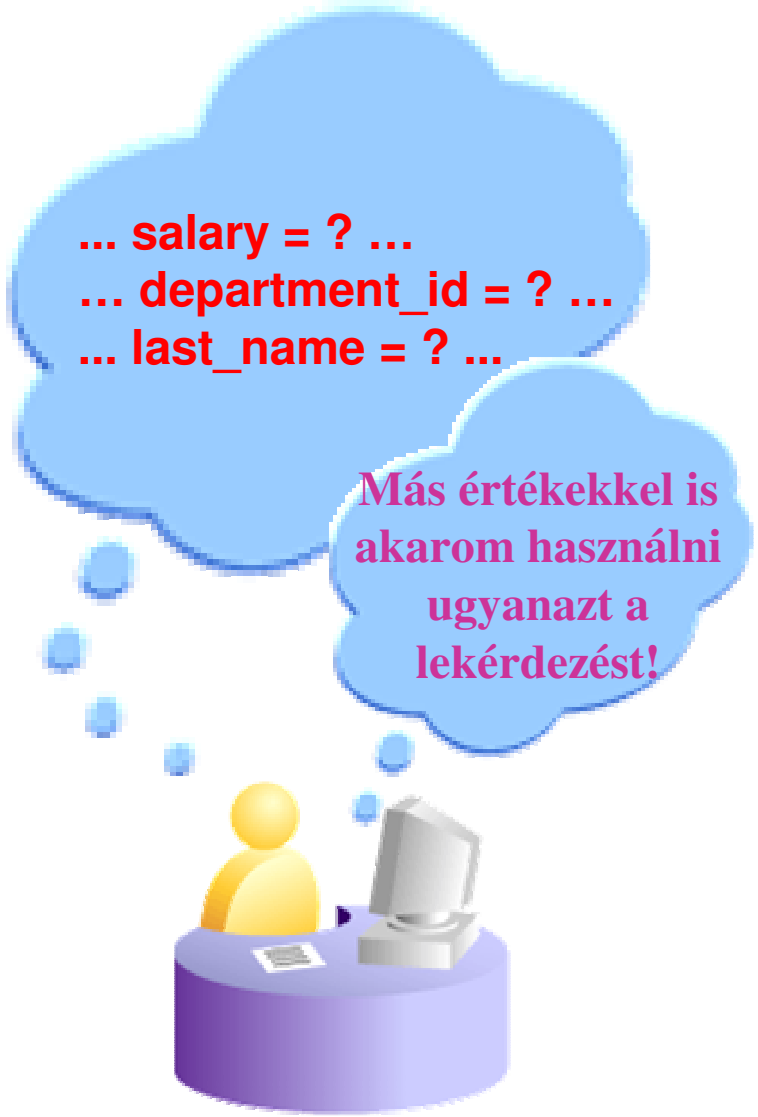
2

- **Az osztály szerint növekvő és azon belül fizetés szerint csökkenő rendezés:**

```
SELECT last_name, department_id, salary
FROM   employees
ORDER BY department_id, salary DESC;
```

3

Változók helyettesítése futási időben



Az *i*SQL*Plus-ban lehetséges helyettesítő változókat definiálni.

A helyettesítő változók futási időben kérnek értéket.

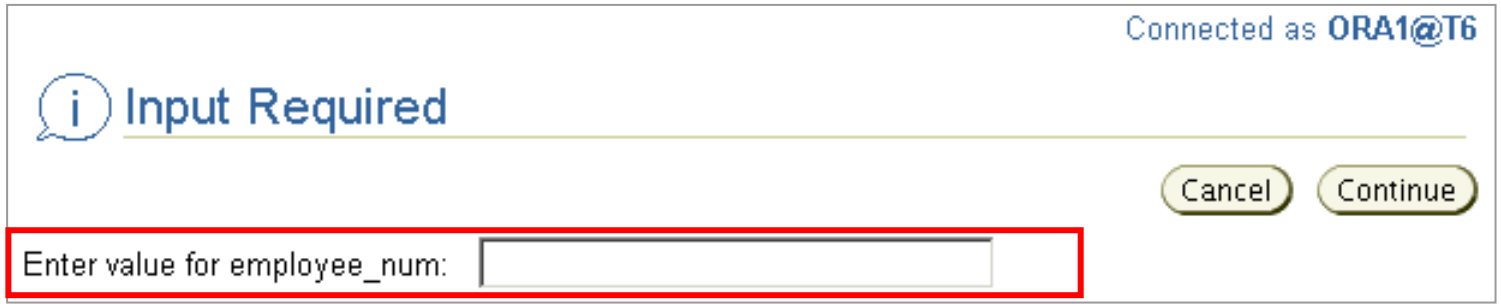
A helyettesítő változók használata

- Az *i*SQL*Plus-ban kétféleképp lehet helyettesítő változó használni:
  - `&változó` esetén futáskor a bekért értéket behelyettesíti, majd a következő előfordulásnál újra értéket vár.
  - `&&változó` esetén definiálja a változót, a bekért értéket adja neki, és a kilépésig, vagy az `undefine változó` feloldó utasításig ez a `&&változó` és `&változó` értéke.
- A helyettesítések tipikus használata:
  - `WHERE` feltételek
  - `ORDER BY` oszloplista
  - Oszlopkifejezések
  - Táblanevek
  - Teljes `SELECT` utasítások

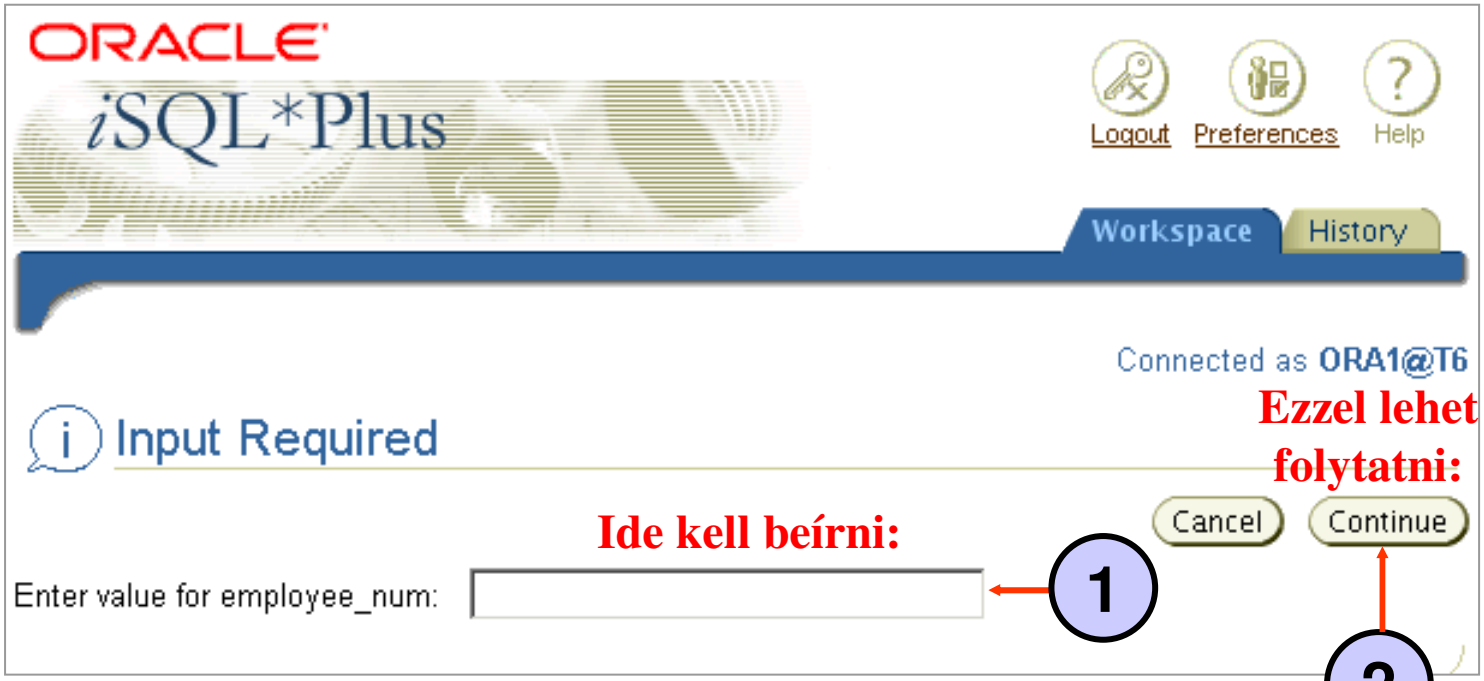
A &változó helyettesítés használata

Kérjük be a dolgozó azonosítóját (&employee_num), majd listázzuk ennek a dolgozónak az adatait:

```
SELECT employee_id, last_name, salary, department_id
FROM   employees
WHERE  employee_id = &employee_num ;
```



A &változó helyettesítés használata



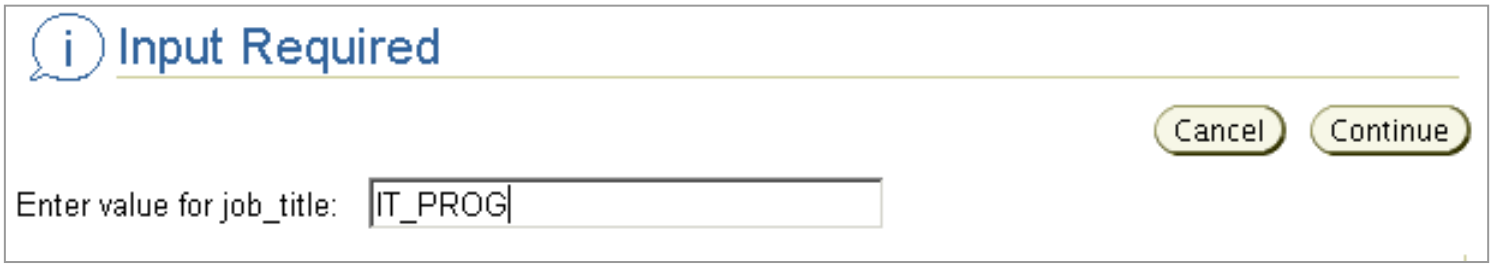

```
old 3: WHERE employee_id = &employee_num
new 3: WHERE employee_id = 101
```

EMPLOYEE_ID	LAST_NAME	SALARY	DEPARTMENT_ID
101	Kochhar	17000	90

Karakterek és dátumértékek megadása helyettesítő változókkal

Dátumtípus és karaktertípus esetén egyszeres idézőjelek közé kell tenni a változót!

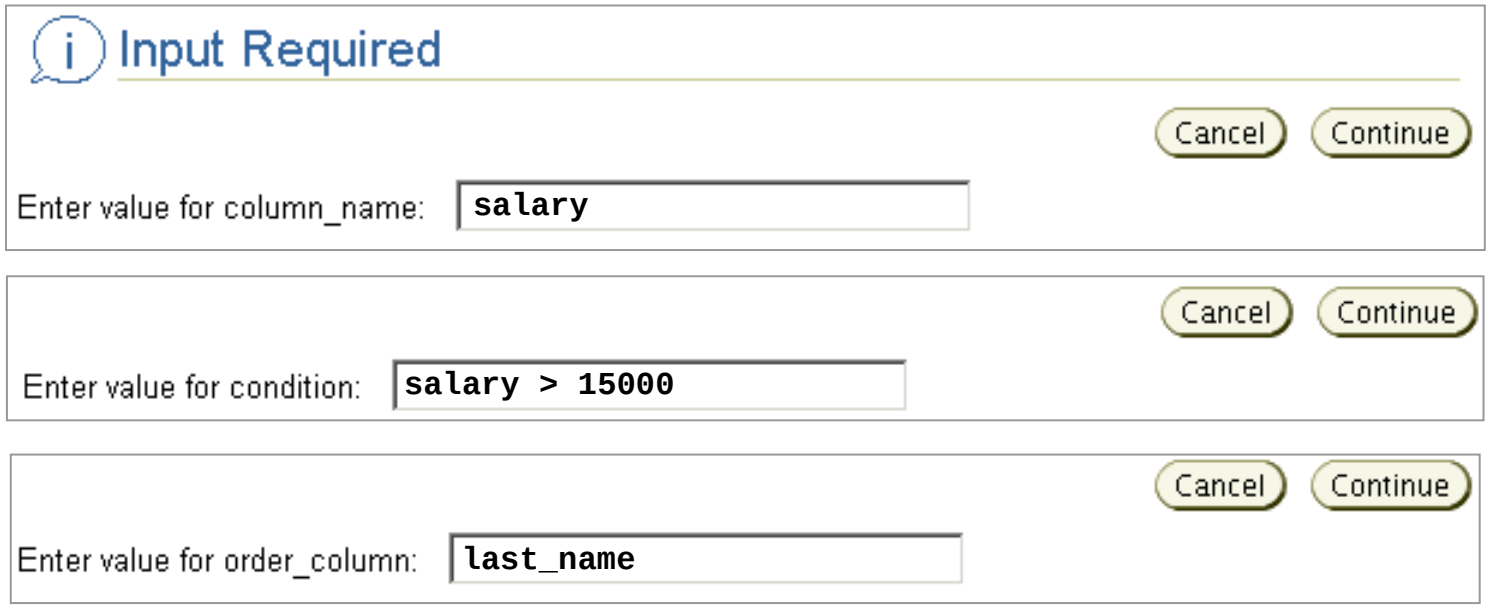
```
SELECT last_name, department_id, salary*12
FROM   employees
WHERE  job_id = '&job_title' ;
```



LAST_NAME	DEPARTMENT_ID	SALARY*12
Hunold	60	108000
Ernst	60	72000
Lorentz	60	50400

Oszlopnevek, szűrő feltételek, szövegrészek megadása

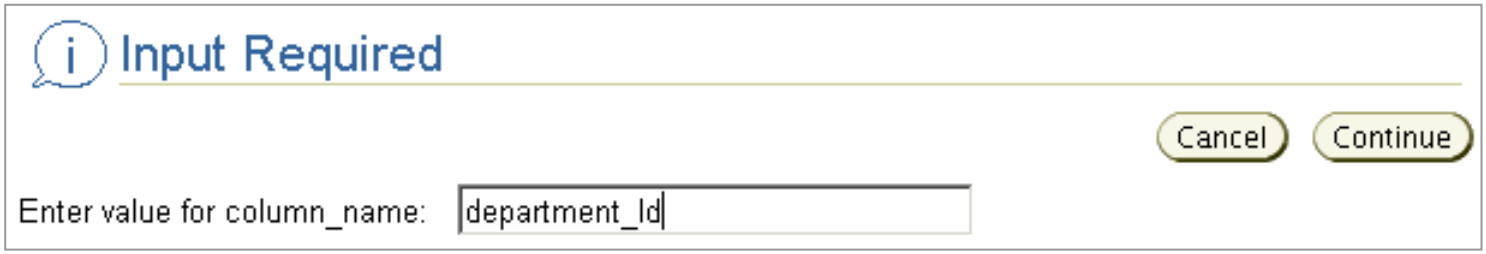
```
SELECT employee_id, last_name, job_id,&column_name
FROM   employees
WHERE  &condition
ORDER BY &order_column ;
```



A &&változó helyettesítés használata

Ha a beolvasott értéket többször is használni akarjuk, akkor **&&változó** módon kérjük be az értéket:

```
SELECT   employee_id, last_name, job_id, &&column_name
FROM     employees
ORDER BY &column_name ;
```



EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
200	Whalen	AD_ASST	10
201	Hartstein	MK_MAN	20

...

20 rows selected.

A hozzárendelés megszüntethető az **UNDEFINE column_name** utasítással.

Az *i*SQL*Plus DEFINE utasítása

- Az *i*SQL*Plus-ban **DEFINE** utasítással is lehet változót definiálni és értéket adni.
- A változót megszűntetni az *i*SQL*Plus-ban **UNDEFINE** utasítással lehet.

```
DEFINE employee_num = 200

SELECT employee_id, last_name, salary, department_id
FROM   employees
WHERE  employee_id = &employee_num ;

UNDEFINE employee_num
```

A VERIFY utasítás

A **VERIFY** kapcsolóval lehet megjeleníteni a helyettesítés előtti és utáni értéket:

```
SET VERIFY ON
SELECT employee_id, last_name, salary, department_id
FROM   employees
WHERE  employee_id = &employee_num;
```


```
old   3: WHERE  employee_id = &employee_num
new   3: WHERE  employee_id = 200
```

A rendszerváltozók beállítását a **SHOW ALL** utasítással ellenőrizhetjük.

Összefoglalás

Ebben a részben megtanultuk:

- hogyan lehet szűrni a **WHERE** feltétellel:
  - az összehasonlító feltételek használatával,
  - a **BETWEEN**, **IN**, **LIKE** és **NULL** feltételek használatával,
  - a logikai **AND**, **OR** és **NOT** műveletek alkalmazásával,
- hogyan lehet rendezni **ORDER BY** résszel.

```
SELECT  *|{[DISTINCT] column|expression [alias],...}
FROM    table
[WHERE  condition(s)]
[ORDER BY {column, expr, alias} [ASC|DESC]] ;
```

- hogyan lehet futási időben megadni a szűrő vagy rendező feltételt a helyettesítő változók segítségével *i*SQL*Plus-ban.

Feladatok

1. Adjuk meg a vezetéknevét (last_name) és fizetését (salary) azoknak a dolgozóknak, akik fizetése 12000 dollárnál több! Mentsük el a megoldást a `lab_02_01.sql` állományba!
2. Adjuk meg a vezetéknevét (last_name) és osztályát (department_id) annak a dolgozónak, akinek az azonosítója 176!
3. Módosítsuk a `lab_02_01.sql` állományt úgy, hogy azokat listázza ki, akiknek a fizetése nem esik 5000 és 12000 dollár közé! Mentsük el a megoldást a `lab_02_03.sql fájlba`.

Feladatok

4. Adjuk meg a vezetéknevét (last_name), beosztását (job_ID), és belépési dátumát (hire_date) azoknak a dolgozóknak, akiknek a vezetékneve vagy Matos vagy Taylor! Rendezzük az eredmény a belépési dátum (hire_date) szerint növekvő sorrendben!
5. Adjuk meg a vezetéknevét (last_name) és osztályszámát (department_id) azoknak a dolgozóknak, akiknek az osztályszámuk 20 vagy 50! Rendezzük az eredményt a vezetéknév szerint!
6. Módosítsuk a lab_02_03.sql állományt úgy, hogy azoknak a vezetéknevét (last_name) és fizetését (salary) listázza ki, akik fizetése 5000 és 12000 közé esik és a 20 vagy 50 sorszámú (department_id) osztályon dolgoznak! Nevezzük át az oszlopokat dolgozókra (Employee), illetve havi fizetésre (Monthly Salary)! Mentsük el a megoldást a lab_02_06.sql állományba!

Feladatok

7. Adjuk meg azoknak a vezetéknevét (last_name) és belépési dátumát (hire_date), akik 1994-ben léptek be!
8. Adjuk meg azoknak a vezetéknevét (last_name) és beosztását (job title), akiknek nincs főnökük (manager)!
9. Adjuk meg azoknak a vezetéknevét (last_name), fizetését (salary) és jutalékát (commission), akiknek van jutalékuk! Rendezzük az eredményt a fizetés, és azon belül a jutalék szerint csökkenő sorrendben!
10. Módosítsuk az 1. feladatot úgy, hogy futási időben lehessen megadni azt az értéket, aminél többet kereső dolgozók vezetéknevét és fizetését keressük! Mentsük el a megoldást a `lab_02_10.sql` állományba! Futassuk le úgy, hogy 12000 legyen a beadott érték!

Feladatok

11. Adjunk meg egy olyan lekérdezést, amely bekéri a főnök azonosítóját (manager_ID), és hogy melyik oszlop szerint akarjuk rendezni az eredményt, és megadja azoknak a dolgozóknak az azonosítóját (employee_ID), vezetéknevét (last_name), fizetését (salary), és osztályát (department_id), akiknek a megadott számú dolgozó a főnökük! Teszteljük a következő kombinációkkal:

a) manager ID = 103, vezetéknév (last_name) szerint rendezve,

b) manager ID = 201, fizetés (salary) szerint rendezve,

c) manager ID = 124, dolgozóazonosító (employee_ID) szerint rendezve!

Feladatok

12. Adjuk meg azoknak a dolgozóknak a vezetéknevét (last_name), akiknek a vezetéknevében a harmadik betű *a*!
13. Adjuk meg azoknak a dolgozóknak a vezetéknevét (last_name), akiknek a vezetéknevében van *a* és *e* betű is!
14. Adjuk meg azoknak a dolgozóknak a vezetéknevét (last_name), beosztását (job_id), és fizetését (salary), akiknek a beosztása vagy képviselő (sa_rep) vagy tőzsdeügynök (st_clerk) és a fizetésük nem egyenlő sem 2500-zal, se 3500-zal, se 7000-rel!
15. Módosítsuk a `lab_02_06.sql` állományt úgy, hogy azoknak a vezetéknevét (last_name), fizetését (salary) és jutalékát (commission) listázza ki, akiknek a jutaléka 20%! Mentsük el a megoldást a `lab_02_15.sql` állományba!