Práca s NULL a množinové operácie

1) **Trojhodnotová logika**
   a) **Unknown**
   b) **NULL**
   c) **SET ANSI_NULL ON | OFF**
2) **Množinové operácie**
   a) **UNION [ALL]**
   b) **INTERSECT**
   c) **EXCEPT**
3) **Príklady**

1) Trojhodnotová logika

a) Neznáme (UNKNOWN ≠ NULL)

OR	TRUE	FALSE	UNKNOWN
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	UNKNOWN
UNKNOWN	TRUE	UNKNOVN	UNKNOWN

AND	TRUE	FALSE	UNKNOWN
TRUE	TRUE	FALSE	UNKNOWN
FALSE	FALSE	FALSE	FALSE
UNKNOWN	UNKNOWN	FALSE	UNKNOWN

	NOT
TRUE	FALSE
FALSE	TRUE
UNKNOWN	UNKNOWN

V teórii relačných DB výsledok porovnanie skutočných hodnôt s UNKNOWN vychádza z definície OR a AND.

b) NULL

SQL používa **NULL** reprezentáciu ***chýbajúcich*** hodnôt v DB, teda údaj ***neexistuje***.
V reálnych DB systémoch výsledok výrazu, ktorý obsahuje NULL je NULL. Presnejšie: http://www.microsoft.com/technet/prodtechnol/sql/70/books/relation.mspx

OR	TRUE	FALSE	Null
TRUE	TRUE	TRUE	Null
FALSE	TRUE	FALSE	Null
Null	Null	Null	Null

AND	TRUE	FALSE	Null
TRUE	TRUE	FALSE	Null
FALSE	FALSE	FALSE	Null
Null	Null	Null	Null

=	TRUE	FALSE	Null
TRUE	TRUE	FALSE	Null
FALSE	FALSE	TRUE	Null
Null	Null	Null	Null - Default

	IS NULL	IS NOT NULL
ine	FALSE	TRUE
NULL	TRUE	FALSE

c)

```
SET ANSI_NULLS OFF;
 =>
Null = Null sa vyhodnotí ako True
Null = <ine> sa vyhodnotí ako False
```

http://www.databasejournal.com/features/mssql/article.php/3399931

Príklady

```
USE tempdb;
GO

IF OBJECT_ID('S') IS NOT NULL DROP TABLE S -- Stlpec
GO

CREATE TABLE S(id INT)
GO

INSERT S VALUES(0);
INSERT S VALUES(NULL);
INSERT S VALUES(1);

SELECT * FROM S s

-- 1)
--SET ANSI_NULLS ON; -- DEFAULT JE ON
SELECT * FROM S WHERE id = NULL;
SELECT * FROM S WHERE id <> NULL;

SET ANSI_NULLS OFF;
SELECT * FROM S WHERE id = NULL;
SELECT * FROM S WHERE id <> NULL;
SET ANSI_NULLS ON; -- DEFAULT JE ON

-- 2)
-- IS (NOT) NULL nezavisi od ANSI_NULLS:
SELECT * FROM S WHERE id IS NULL;
SELECT * FROM S WHERE id IS NOT NULL;

-- vysledok tiez nezavisi od ANSI_NULLS
SELECT * FROM S WHERE id <> 0; -- 1 ale NULL nikdy

-- 3)
--SET ANSI_NULLS OFF;
SELECT id, CASE WHEN id = NULL THEN 'NIKDY'
                WHEN id =    0 THEN 'nula'
                WHEN id =    1 THEN 'jedna'
           END
FROM S;
--SET ANSI_NULLS ON;

SELECT id, CASE WHEN id IS NULL THEN 'NIKDY'
                WHEN id =     0 THEN 'nula'
                WHEN id =     1 THEN 'jedna'
           END
FROM S;
```

```sql
-- 4a)
SELECT s1.id, s2.id FROM S s1 JOIN S s2
--                       ON s2.id = s1.id
                         ON (s2.id = s1.id) OR ( s1.id IS NULL
                                                 AND s2.id IS NULL )


-- 4b)
SELECT 5*NULL
SELECT 5/NULL
SELECT 5/0

-- 5)
-- Vratme tie mesacne prijmy, ktore sa nerovnaju 10-nasobku poplatku
-- v tabulke navstevy
-- (pretoze poplatky su ovela nizsie ako prijmy a su dve NULL poplatky,
--  mali by sme obdrzat 7 = 10-3 )
--  NO:
SELECT * FROM Pacienti p
WHERE p.mesPrijem NOT IN(
     SELECT 10*n.poplatok FROM Navstevy n)

SELECT * FROM Pacienti p
WHERE p.mesPrijem NOT IN(
     SELECT 10*n.poplatok FROM Navstevy n
     WHERE n.poplatok IS NOT NULL )

-- alebo:
SELECT * FROM Pacienti p
WHERE p.mesPrijem <> ALL
     (
          SELECT 10*n.poplatok FROM Navstevy n
          WHERE n.poplatok IS NOT NULL )
```

2) Množinové operácie

Množinové operácie v SQL spájajú dva alebo viac dopytov s kompatibilnými výsledkami.
Výhodou množinových operácií je zjednodušenie, resp. sprehľadnenie zložitejších dopytov. Ich **nevýhodou** môže byť menej optimálny kód, beh ktorého trvá dlhšie, veď štandardne riadky tabuľky treba preskenovať, prejsť dvakrát, raz pre každý *Select* operand, a v prípade veľkých tabuliek to môže znamenať časovú stratu.

a) UNION [ALL]

UNION a UNION ALL operátory umožňujú spojiť viac výsledkov (dopytov) do jedného. Na rozdiel od JOIN, ktorý predovšetkým používame na spojenie stĺpcov (a pochopiteľne aj riadkov), UNION sa používa na **spojenie riadkov**, pritom:

- počet a poradie stĺpcov musia byť rovnaké
- dátové typy zodpovedajúcich stĺpcov musia byť kompatibilné

UNION ALL na rozdiel od UNION vráti aj duplicitné riadky.

```
USE Poliklinika;
GO

SELECT p.krstne, p.idP, 'P' typ FROM Pacienti p
UNION -- ALL  -- 15 / 15
SELECT L.krstne, L.idL, 'L' typ FROM Lekari L

SELECT p.mesPrijem FROM Pacienti p
UNION -- ALL  -- 18 / 32=10+22
SELECT n.poplatok FROM Navstevy n
```

b) INTERSECT

Operátor INTERSECT porovnáva výsledky viac SELECT príkazov a vráti DISTINCT hodnoty.

```
SELECT p.krstne FROM Pacienti p -- 10
INTERSECT -- 2
SELECT L.krstne FROM Lekari L

-- Ako reaguje na duplicitne hodnoty:
INSERT Pacienti(idP, krstne)
VALUES(100, 'Klara')
INSERT Lekari(idL, krstne)
VALUES(200, 'Klara')

SELECT p.krstne FROM Pacienti p -- 10
INTERSECT -- 2
```

```
SELECT L.krstne FROM Lekari L

DELETE FROM Pacienti WHERE idP = 100 -- NO alias !!!!!!!!!!!
DELETE FROM Lekari WHERE idL = 200

-- Reakcia na NULL - nic zvlastne, lebo vrati distinct a teda jeden NULL
SELECT p.mesPrijem FROM Pacienti p -- 10
INTERSECT -- 1
SELECT n.poplatok FROM Navstevy n
```

c) EXCEPT

Operátor EXCEPT porovnáva výsledky viac SELECT príkazov a vráti DISTINCT hodnoty.
Na rozdiel od INTERSECT nie je symetrická operácia – záleží na poradí.

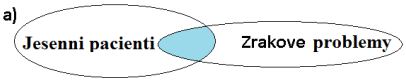
(INTERSECT má vyššiu prioritu ako EXCEPT)

```
SELECT p.mesPrijem FROM Pacienti p -- 10
EXCEPT -- 7
SELECT 10*n.poplatok FROM Navstevy n -- 22
 -- WHERE n.poplatok IS NOT NULL -- 8 !!!
```

3) Príklady

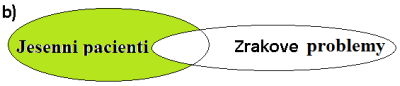
```
SELECT p.krstne FROM Pacienti p -- 10
INTERSECT -- 1
SELECT L.krstne FROM Lekari L -- 5

----
---- a) Akeho lekara (idL)   navstivil pacient s id 2 (Stefan)
---- b) Akeho lekara (idL) NEnavstivil pacient s id 2 (Stefan)
----
SELECT N.idL FROM Navstevy N
     WHERE N.idP = 2

---- Pokracovanie:
SELECT L.idL FROM Lekari L
EXCEPT
SELECT N.idL FROM Navstevy N
     WHERE N.idP = 2
```

a)



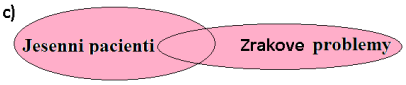
Jesenni pacienti so zrakovymi problemami

b)



Jesenni pacienti bez zrakovych problemov

c)



Jesenni pacienti alebo pacienti so zrak.problemami

a) Jesenní pacienti so zrakovými problémami

```
Select N.idP, N.idL, Month(N.den) from Navstevy N Where Month(N.den) >= 9 -- Jesenni pacienti -- 9
INTERSECT      -- Jesenni pacienti so zrakovymi problemami -- 5
Select N.idP, N.idL, Month(N.den) from Navstevy N join lekari L on n.idL=l.idL
Where L.spec='Ocny' -- Pacienti so srdcovymi problemami -- 9
-- Strucnejsie (kedze mame jedineho ocneho s idL 9):
-- Select N.idP, N.idL, Month(N.den) from Navstevy N Where idL = 1--Pacienti so zrakovymi problemami -- 9
```

b) Jesenní pacienti bez zrakových problémov

```
Select N.idP, N.idL, Month(N.den) from Navstevy N Where Month(N.den) >= 9 -- Jesenni pacienti -- 9
Except         -- Jesenni pacienti bez zrakovych problemov -- 4
Select N.idP, N.idL, Month(N.den) from Navstevy N Where idL = 1 -- Pacienti so zrakovymi problemami -- 9
```

c) Jesenní pacienti alebo pacienti so zrakovými problémami

```
Select N.idP, N.idL, Month(N.den) from Navstevy N Where Month(N.den) >= 9 -- Jesenni pacienti -- 9
Union          -- Jesenni pacienti alebo pacienti so zrak.problemami -- 11
Select N.idP, N.idL, Month(N.den) from Navstevy N Where idL = 1
```

Nižšie uvedieme aj alternatívne riešenia. Usúďte, ktoré verzie sú prehľadnejšie, ale v prípade masívnych tabuliek by boli menej optimálne.

a)

```
Select N.idP, N.idL, Month(N.den) mes From Navstevy N Where Month(N.den) >= 9 -- Jesenni pacienti -- 9
INTERSECT      -- Jesenni pacienti so zrakovymi problemami -- 5
Select N.idP, N.idL, Month(N.den) From Navstevy N Where idL = 1 -- Pacienti so zrakovymi problemami -- 9

Select N1.idP, N1.idL, Month(N1.den) mes From Navstevy N1
Where Month(N1.den) >= 9 AND N1.idL = 1

Select N1.idP, N1.idL, Month(N1.den) mes From Navstevy N1
       Where Month(N1.den) >= 9 AND N1.idL In (Select N1.idL From Navstevy N1
                                               Where N1.idL = 1)
```

b)

```
Select N.idP, N.idL, Month(N.den) mes From Navstevy N Where Month(N.den) >= 9 -- Jesenni pacienti -- 9
Except         -- Jesenni pacienti bez zrakovych problemov -- 4
Select N.idP, N.idL, Month(N.den) From Navstevy N Where idL = 1 -- Pacienti so zrakovymi problemami -- 9

Select N1.idP, N1.idL, Month(N1.den) mes From Navstevy N1
Where Month(N1.den) >= 9 AND N1.idL <> 1

Select N1.idP, N1.idL, Month(N1.den) mes From Navstevy N1
       Where Month(N1.den) >= 9 AND N1.idL NOT In (Select N1.idL From Navstevy N1
                                                   Where N1.idL = 1)
```

c)

```
Select N.idP, N.idL, Month(N.den) mes From Navstevy N Where Month(N.den) >= 9 -- Jesenni pacienti -- 9
Union          -- Jesenni pacienti alebo pacienti so zrak.problemami -- 11
Select N.idP, N.idL, Month(N.den) mes From Navstevy N Where idL = 1--Pacienti so zrakovymi problemami--9

Select N1.idP, N1.idL, Month(N1.den) mes From Navstevy N1
Where Month(N1.den) >= 9 OR N1.idL = 1
Order by N1.idP
```