

SQL Programming

In the Land of NULLs

SQL is unlike most other current programming languages in that it provides a special value known as the NULL value.

This is one of the features of SQL that distinguishes it from most other programming languages, and there is an ongoing debate in db circles as to whether or not the complexity introduced by this feature is worth the trouble.

We'll start this module with a question.

How should missing or unknown values be represented in the database?

In the olden days, back when programmers maintained file systems, the tradition was to assign a value to every field in a file when the record was initialized.

Numeric fields were often initialized to 0 (zero), alphanumeric fields were often initialized to spaces.

One of the primary reasons for doing this was because data fields were strongly 'typed', and if the machine encountered an invalid value in a field (say spaces in a numeric field), the program would ABEND – abnormally end. Right then and there the program would terminate.

And, regardless of the time of day, the operations staff would call the user specialist, or the programmer, and have them come remedy the problem.

So, if errant data values can mess things up, what value could be used?

Programmers don't much care for being called in to work in the dead of night to remedy a failing program, so early on in their careers they learned the value of properly initialized records.

This begs the question, who picks the value to use for these unknown items?

Programmers often selected the values themselves, occasionally there were business standards that dictated what values should be used.

But what value would you use for a name field that hadn't been identified? Or a birthdates field when the true birth date value is unknown?

Programmers often used zeroes in numeric fields when those fields were initialized. But consider the field: HOURLY_RATE. Does a 0 value mean that we don't know the employee's hourly rate, or is it the case that this individual is a VOLUNTEER, and is working in an unpaid position.

Database technology sought to remedy this problem with the introduction of the NULL value.

The issues surrounding proper and efficient use of NULLs have sparked a number of discussions/debates/arguments in DB circles.

For our purposes, the best reason to use NULL is when the data value is unknown.

This will be our rationale / underlying premise for all future discussions regarding NULL.

How do we use this NULL value?

Note: in the slides that follow although I've hard-coded the NULL value into the statements, SQL would behave in the same fashion if it were to encounter the NULL value as a columnar value.

Module 11: NULLs

iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High ...

File Edit View Favorites Tools Help Address u/isqlplus Go

Back Forward Stop Refresh Home Search Favorites

ORACLE iSQL*Plus Password Log Out Help

Script Location: Browse... Load Script

Enter statements:

```
SET NULL <null>;  
  
SELECT 3 + NULL  
FROM dual;
```

Execute Output: Work Screen Clear Screen Save Script

3+NULL	<null>
--------	--------

Done Internet

Page C-2 Math and NULLs

In math, any operation that uses a NULL value begets (propagates) a NULL value. That is, if any part of the operation is NULL, then the result is NULL (ie. I don't know, it's unknown, ...)

I've got some marbles in my left hand, and 2 more marbles in my right hand. After I put them all together, how many have I got?

I don't know? More than 2, but beyond that it's anybody's guess.

Module 11: NULLs

Page C-3 Dates and NULLs

Date operations on a NULL value also beget NULL values.

iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High ...

File Edit View Favorites Tools Help Address u/sqlplus Go

Back Forward Stop Refresh Home Search Favorites

ORACLE iSQL*Plus Password Log Out Help

Script Location: Browse... Load Script

Enter statements:

```
SET NULL <null>;  
  
SELECT TO_DATE(NULL) + 1  
FROM dual;
```

Execute Output: Work Screen Clear Screen Save Script

TO_DATE(N)

<null>

Done Internet

Module 11: NULLs

Page C-4 Characters and NULLs

Character operations (of which there is only one) do NOT beget NULL values.

What an inconsistency!

I guess Oracle was abiding by that old adage: *A foolish consistency is the hobgoblin of little minds.*

iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High ...

File Edit View Favorites Tools Help Address u/isqlplus Go

Back Forward Stop Refresh Home Search Favorites

ORACLE iSQL*Plus Password Log Out Help

Script Location: Browse... Load Script

Enter statements:

```
SET NULL <null>;  
  
SELECT  NULL || ' See what I mean?'  
FROM    dual;
```

Execute Output: Work Screen Clear Screen Save Script

NULL||'SEEWHATIME
See what I mean?

Done Internet

How do NULLs affect functions?

Generally, NULLs propagate NULLs.

Module 11: NULLs

Page D-2 Numeric Functions

NULLs in numeric functions generally return NULL values.

iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High ...

File Edit View Favorites Tools Help Address u/sqlplus Go

Back Forward Stop Refresh Home Search Favorites

ORACLE iSQL*Plus Password Log Out Help

Script Location: Browse... Load Script

Enter statements:

```
SET NULL <null>;

SELECT  MOD(NULL, 3)
FROM    dual;
```

Execute Output: Work Screen Clear Screen Save Script

MOD(NULL,3)
<null>

Done Internet

Module 11: NULLs

Page D-3 Character Functions

NULLs propagate NULLs in character functions.

At least we're approaching consistency now.

iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High ...

File Edit View Favorites Tools Help Address u/isqlplus Go

Back Forward Stop Refresh Home Search Favorites

ORACLE iSQL*Plus Password Log Out Help

Script Location: Browse... Load Script

Enter statements:

```
SET NULL <null>;  
  
SELECT  UPPER (NULL)  
FROM    dual;
```

Execute Output: Work Screen Clear Screen Save Script

U
<null>

Done Internet

Module 11: NULLs

Page D-4 Date Functions

NULLs beget NULLs in date functions.

But you should note, that this particular function is a vendor (Oracle) extension to the SQL language. I'm just using it 'cause it was handy.

iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High ...

File Edit View Favorites Tools Help Address u/isqlplus Go

Back Forward Stop Refresh Home Search Favorites

ORACLE iSQL*Plus Password Log Out Help

Script Location: Browse... Load Script

Enter statements:

```
SET NULL <null>;  
  
SELECT LAST_DAY (TO_DATE (NULL) )  
FROM dual;
```

Execute Output: Work Screen Clear Screen Save Script

LAST_DAY(
<null>

Done Internet

Module 11: NULLs

Page D-5 Intervals and NULLS

I trust you're getting the picture?

iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High Speed Internet

File Edit View Favorites Tools Help Address <http://cisdb02.msje.edu/isql> Go

Back Forward Stop Refresh Home Search Favorites History Mail

ORACLE iSQL*Plus [Password](#) [Log Out](#) [Help](#)

Script Location: Browse... Load Script

Enter statements:

```
SET NULL <null>;

SELECT (CURRENT_DATE - TO_DATE(NULL)) YEAR to MONTH
FROM talent;
```

Execute Output: Work Screen Clear Screen Save Script

(CURRENT_DATE-TO_DATE(NULL))YEARTOMONTH
<null>
<null>
<null>
<null>
<null>

Done Internet

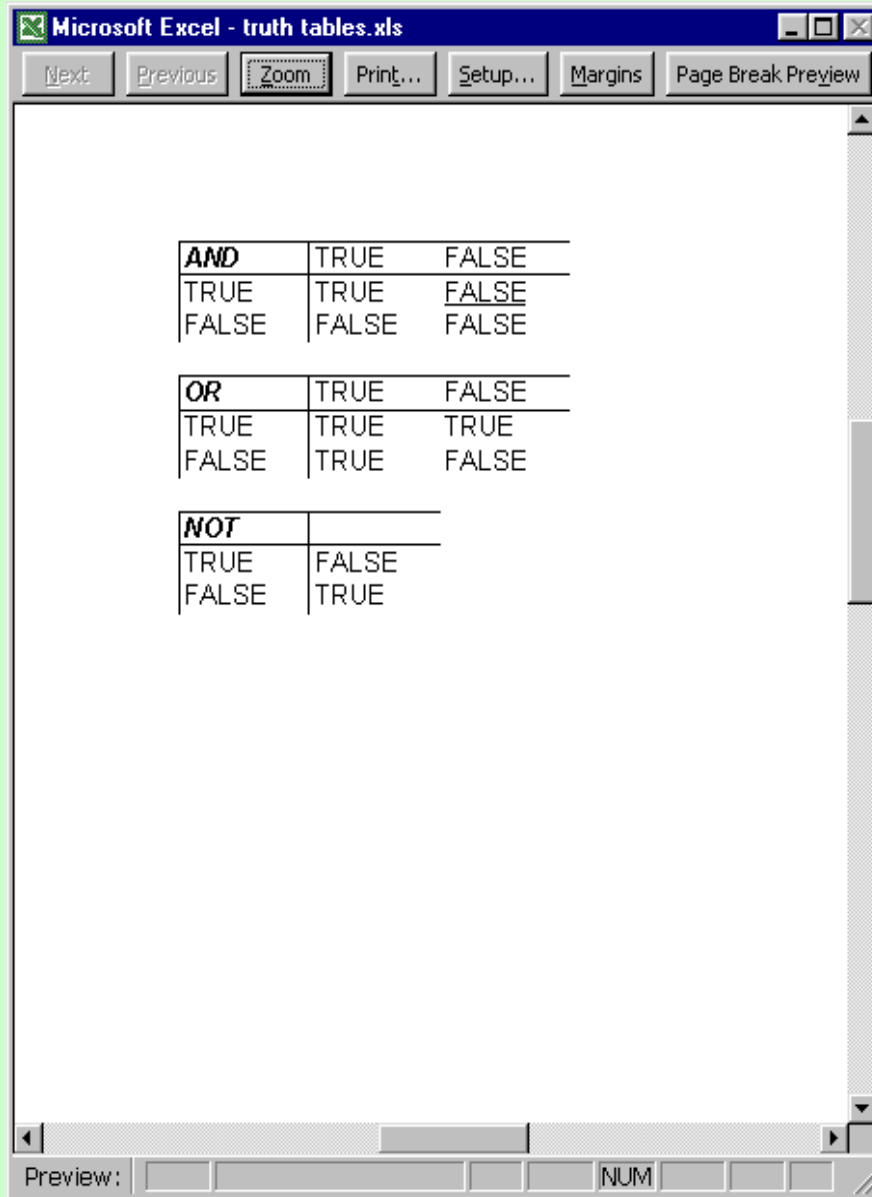
We have the same problem with comparisons (predicates) that we did with operations, that is, how can SQL resolve any predicate condition that contains an unknown value?

In general term, SQL can't resolve unknowns, and, when unknowns creep into the equation, the 2-state logic that most programmers are comfortable with becomes a 3-state or a 3-value logic that permits:

TRUE FALSE UNKNOWN

This 3-value logic tends to complicate things, just a bit.

Module 11: NULLs



The screenshot shows a Microsoft Excel window titled "Microsoft Excel - truth tables.xls". The window contains three truth tables for logical operators. The first table is for the AND operator, the second for the OR operator, and the third for the NOT operator. Each table has a header row and two data rows. The AND table shows that the result is only TRUE when both operands are TRUE. The OR table shows that the result is TRUE if at least one operand is TRUE. The NOT table shows that the result is the opposite of the operand.

AND	TRUE	FALSE
TRUE	TRUE	FALSE
FALSE	FALSE	FALSE

OR	TRUE	FALSE
TRUE	TRUE	TRUE
FALSE	TRUE	FALSE

NOT	
TRUE	FALSE
FALSE	TRUE

Page E-2 Truth Tables

Truth tables are a common device used to describe the logic employed in evaluating a predicate.

Both AND and OR are binary Boolean operators and take two operands. Read down the column under the Boolean for the value of the first operand, and read across the table for the second operand. Where those two values intersect in the table shows how the compound condition will be evaluated.

Consider this WHERE clause:

WHERE (home_country = 'USA' AND theater = 'YES')

And a row in the table containing these values:

home_country = 'USA'

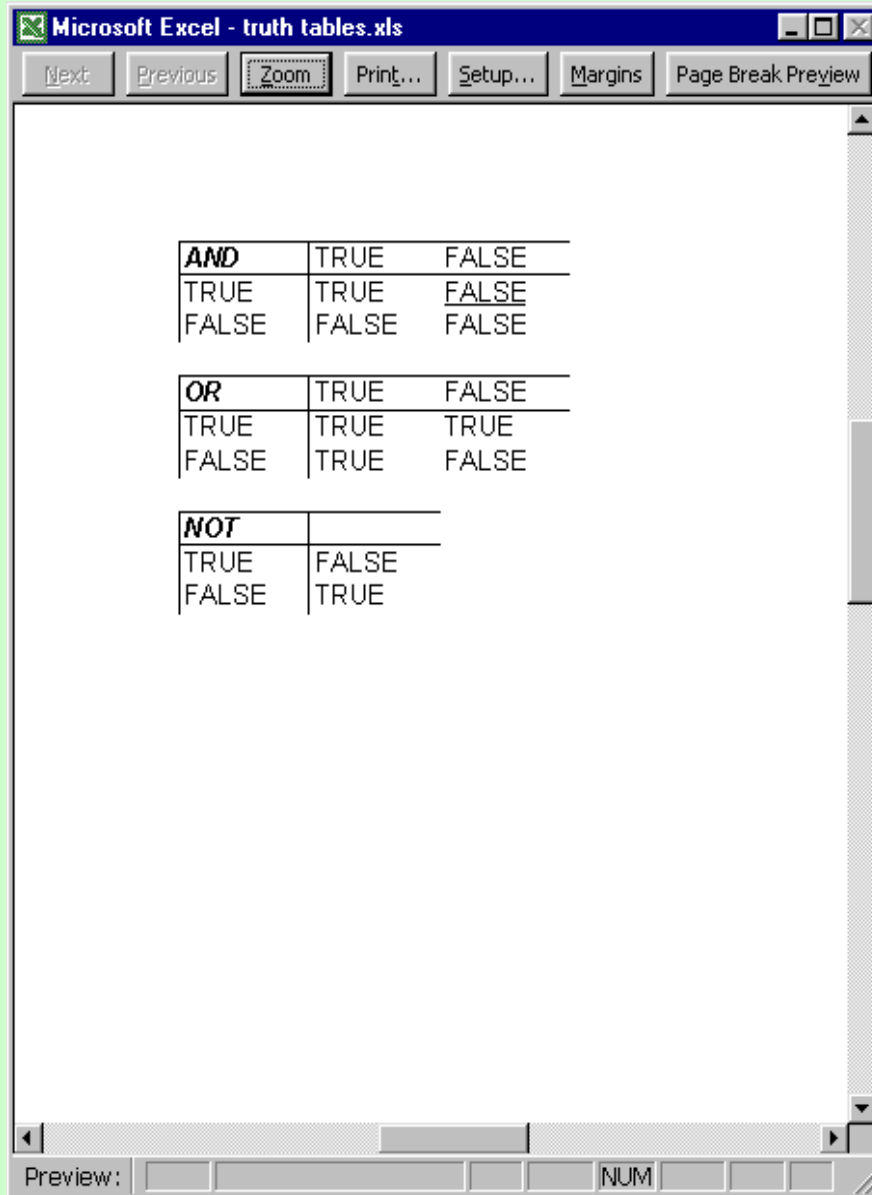
theater = 'NO'

We can see that the first expression evaluates TRUE, while the second expression evaluates FALSE. And what the truth table shows us is how the BOOLEAN AND operation will be evaluated given those two values. The first expression evaluates TRUE (go down the table one row), and the second expression evaluates FALSE (go across the table two columns). Where they intersect (FALSE) is how the AND compound condition will be evaluated.

Module 11: NULLs

Page E-3 Truth Tables (cont)

Take a moment to review these tables and demonstrate for yourself that you understand how these operations will be evaluated.



The screenshot shows a Microsoft Excel window titled "Microsoft Excel - truth tables.xls". The window has a menu bar with "Next", "Previous", "Zoom", "Print...", "Setup...", "Margins", and "Page Break Preview". The main area displays three truth tables. The first table is for the AND operation, the second for the OR operation, and the third for the NOT operation. The status bar at the bottom shows "Preview:" followed by a grid of cells, with the last cell containing the text "NUM".

AND	TRUE	FALSE
TRUE	TRUE	FALSE
FALSE	FALSE	FALSE

OR	TRUE	FALSE
TRUE	TRUE	TRUE
FALSE	TRUE	FALSE

NOT	
TRUE	FALSE
FALSE	TRUE

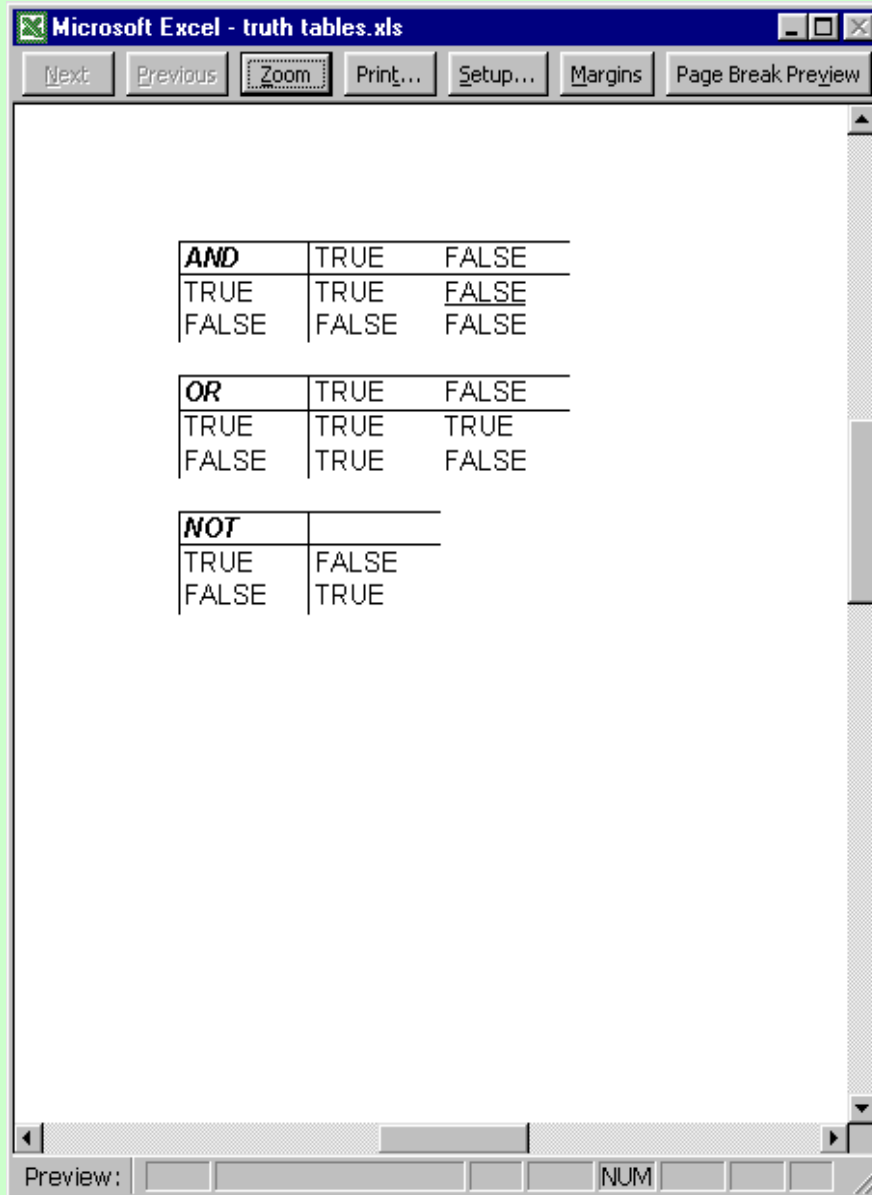
Module 11: NULLs

Page E-4 Simplicity – in a sense

These tables demonstrate a two-state logic in which the only possible outcomes are either TRUE or FALSE.

What happens when we introduce that third value UNKNOWN?

Review the truth tables on the following slide.



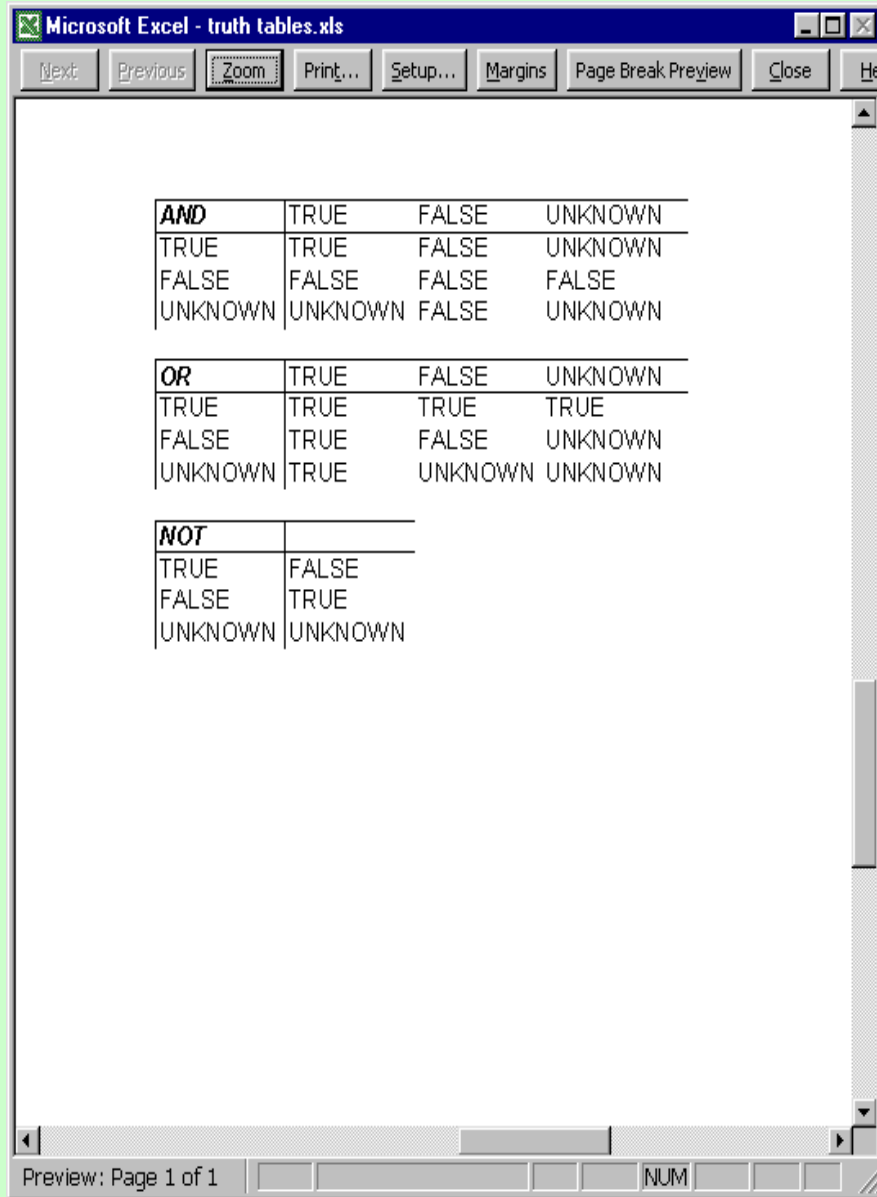
The screenshot shows a Microsoft Excel window titled "Microsoft Excel - truth tables.xls". The window contains three truth tables for logical operations. The first table is for the AND operation, the second for the OR operation, and the third for the NOT operation. Each table has a header row and two data rows. The AND table shows that the result is TRUE only when both inputs are TRUE. The OR table shows that the result is TRUE if at least one input is TRUE. The NOT table shows that the result is the opposite of the input.

AND	TRUE	FALSE
TRUE	TRUE	FALSE
FALSE	FALSE	FALSE

OR	TRUE	FALSE
TRUE	TRUE	TRUE
FALSE	TRUE	FALSE

NOT	
TRUE	FALSE
FALSE	TRUE

Module 11: NULLs



The screenshot shows a Microsoft Excel window titled "truth tables.xls". The window contains three truth tables for logical operations. The first table is for the AND operation, the second for the OR operation, and the third for the NOT operation. Each table has columns for the inputs and the output. The AND table shows that the output is TRUE only when both inputs are TRUE. The OR table shows that the output is TRUE when at least one input is TRUE. The NOT table shows that the output is the opposite of the input.

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

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

NOT	
TRUE	FALSE
FALSE	TRUE
UNKNOWN	UNKNOWN

Page E-5 Three-Value Logic

I hope after reading the tables that they appeal to your intuitions.

With the AND operation, we only get TRUE when both expressions are TRUE. If either is FALSE, we know we have FALSE.

Do you see how only one TRUE value is possible in the AND table?

Do you see how we get that one row and that one column of FALSE?

With the OR operation, if either expression is TRUE, we get TRUE.

Do you understand how we get that one column and that one row of TRUE?

Do you understand why we only have one FALSE?

With the NOT operation, does it make sense that if you don't know what you've got, you don't know what you haven't got? ☺

A NULL value is not EQUAL TO any value.

Nor is a NULL value GREATER THAN any value.

Nor is a NULL value LESS THAN any value.

Two NULL values are neither EQUAL TO, nor NOT EQUAL TO each other.

Now I know what you're thinking:

'Sure whatever you say. But as a programmer I've got a problem. How can I test for NULL values if they're not equal to anything (and NOT not equal to anything)?'

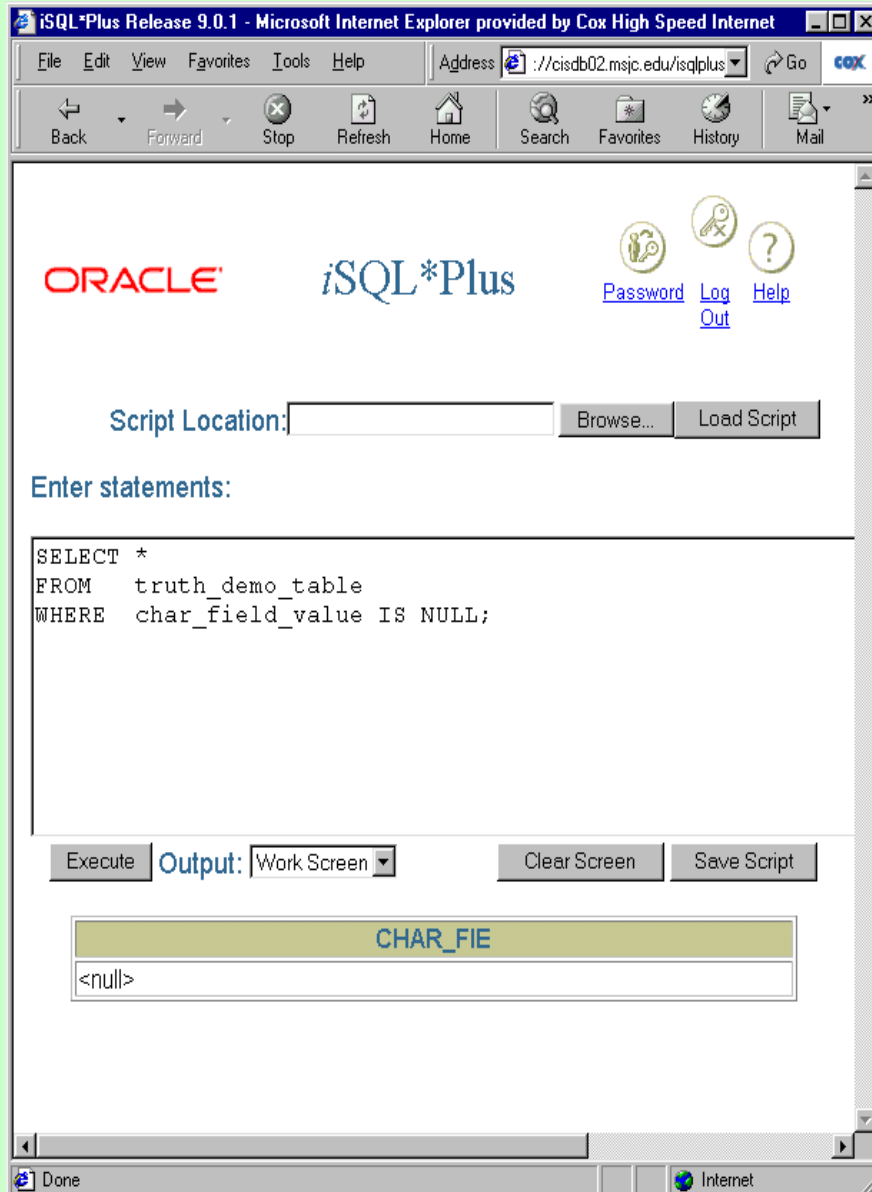
The only way around this catch-22 in SQL is to use a new comparison verb (aka comparison operator) and that's the "IS NULL" verb.

Module 11: NULLs

Page F-2 IS NULL Comparisons

We can find all rows having a NULL value in the char_field_value column with this SELECT statement:

```
SELECT *  
FROM   truth_demo_table  
WHERE  (char_field_value IS NULL);
```



Module 11: NULLs

Page F-3 IS NULL Comparisons

Similarly, we can find all rows that do not contain a NULL value in that column with a program that negates the predicate that we used in the previous program:

```
SELECT *  
FROM truth_demo_table  
WHERE NOT(char_field_value IS NULL);
```

iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High Speed Internet

File Edit View Favorites Tools Help Address ://cisdb02.msje.edu/isqlplus Go

Back Forward Stop Refresh Home Search Favorites History Mail

ORACLE iSQL*Plus Password Log Out Help

Script Location: Browse... Load Script

Enter statements:

```
SELECT *  
FROM truth_demo_table  
WHERE NOT(char_field_value IS NULL);
```

Execute Output: Work Screen Clear Screen Save Script

CHAR_FIE
YES
NO

Done Internet

Module 11: NULLs

Page F-4 IS NULL Comparisons

And for whatever reason (perhaps to read, or sound better), SQL allows us, in this rare circumstance, to embed the NOT keyword in the phrase: IS NOT NULL.

Cast your vote on the Bb site for your preferred method of testing for 'not null'.

iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High Speed Internet

File Edit View Favorites Tools Help Address ://cisdb02.msje.edu/isqlplus Go

Back Forward Stop Refresh Home Search Favorites History Mail

ORACLE iSQL*Plus Password Log Out Help

Script Location: Browse... Load Script

Enter statements:

```
SELECT *
FROM   truth_demo_table
WHERE  char_field_value IS NOT NULL;
```

Execute Output: Work Screen Clear Screen Save Script

CHAR_FIE
YES
NO

Done Internet

Our general rule is that NULLs beget NULLs.

We've seen one exception to this rule already – the character operation of concatenation does NOT propagate NULLs.

Are there any other 'gaps' in our expectations?

SQL treats all NULLs as equivalent when 'gathering them together' under the auspices of a DISTINCT operation, essentially behaving as if NULLs were equal to one another.

Module 11: NULLs

Page H-1 Oracle Extensions

For the examples on the following slides, I'll be using this table which has only three rows:

YES

NO

NULL (the NULL value, not the word null)

The screenshot shows the iSQL*Plus web interface in a Microsoft Internet Explorer browser window. The title bar reads "iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by ...". The browser's menu bar includes File, Edit, View, Favorites, Tools, and Help. The iSQL*Plus logo is displayed, along with links for Password, Log Out, and Help. Below the logo, there is a "Script Location:" field with a "Browse..." button and a "Load S" button. A text area labeled "Enter statements:" contains the SQL query:

```
SELECT *  
FROM   truth_demo_table
```

. Below the text area are buttons for "Execute", "Output:" (with a dropdown menu set to "Work Screen"), "Clear Screen", and "Save Scri". At the bottom, a table displays the results of the query. The table has a single column header "CHAR_FIE" and three rows of data: "YES", "NO", and an empty row.

CHAR_FIE
YES
NO

Module 11: NULLs

Page H-2 NVL Function

Oracle provides a special function to deal with NULL values. This function, the NVL function, returns an alternative value when NULL values are present.

The alternate value must match the column data type, otherwise Oracle-SQL will throw an error.

iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High Speed Internet

File Edit View Favorites Tools Help Address <http://cisdb02.msje.edu/isqlplus> Go

Back Forward Stop Refresh Home Search Favorites History Mail

ORACLE iSQL*Plus [Password](#) [Log Out](#) [Help](#)

Script Location: Browse... Load Script

Enter statements:

```
SELECT NVL(char_field_value, 'Nothing')
FROM truth_demo_table;
```

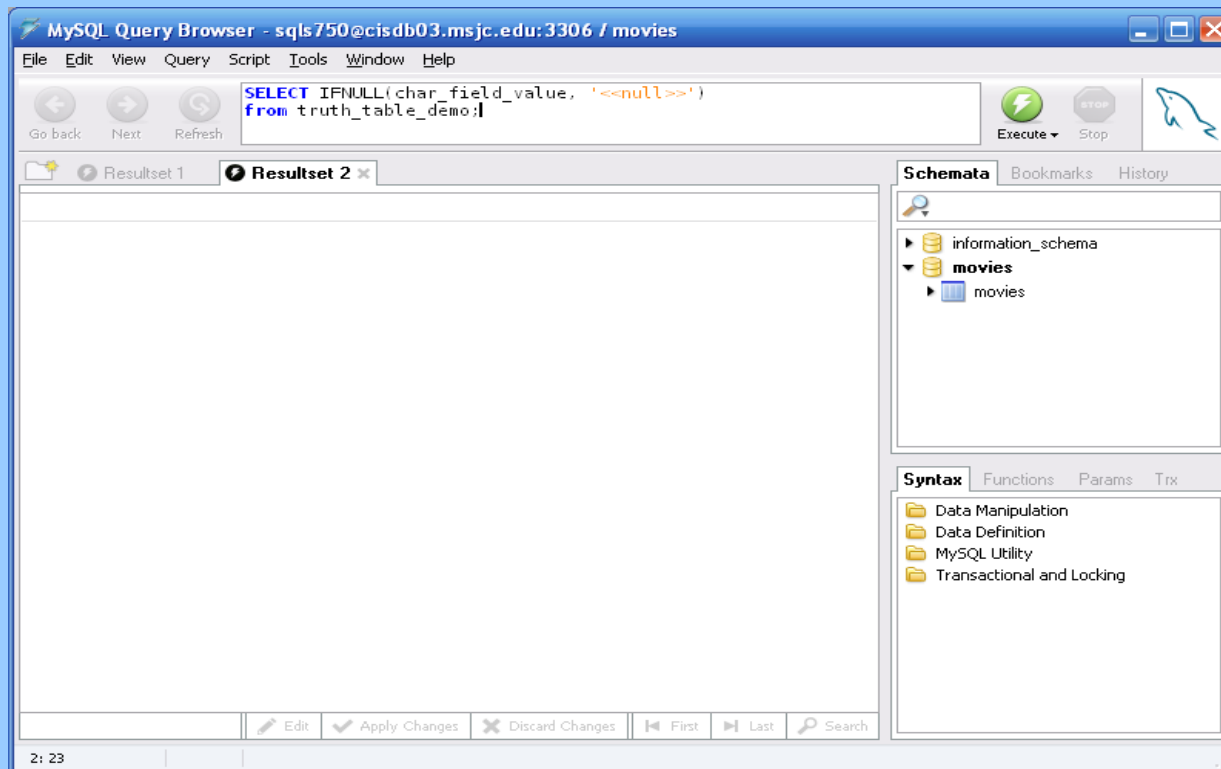
Execute Output: Work Screen Clear Screen Save Script

NVL(CHAR)
YES
NO
Nothing

Done Internet

MySQL uses the IFNULL() function to do pretty much the same thing.

But do note: in MySQL the alternate, or replacement value, does NOT have to be of the same data type as the column data type.



Module 11: NULLs

Page H-3 ORDER BY Feature

The ORDER BY clause includes a phrase that allows the programmer to indicate where/how NULLS are to be sorted.

They may appear first in the list (NULLS FIRST), or last in the listing (NULLS LAST).

In MySQL NULLs are treated as a 'low' value, and in an ASC sort they are presented first, and they appear last in the list if you specify DESC (to sort in descending order).

iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High Speed Internet

File Edit View Favorites Tools Help Address http://cisdb02.msje.edu/isqlplus Go

Back Forward Stop Refresh Home Search Favorites History Mail

ORACLE iSQL*Plus Password Log Out Help

Script Location: Browse... Load Script

Enter statements:

```
SET NULL <null>;

SELECT      *
FROM        truth_demo_table
ORDER BY    char_field_value ASC NULLS FIRST;
```

Execute Output: Work Screen Clear Screen Save Script

CHAR_FIE
<null>
NO
YES

Done Internet

NULL

TRUE, FALSE, UNKNOWN

2-state logic

3-state logic, 3-value logic

Truth tables

IS NULL

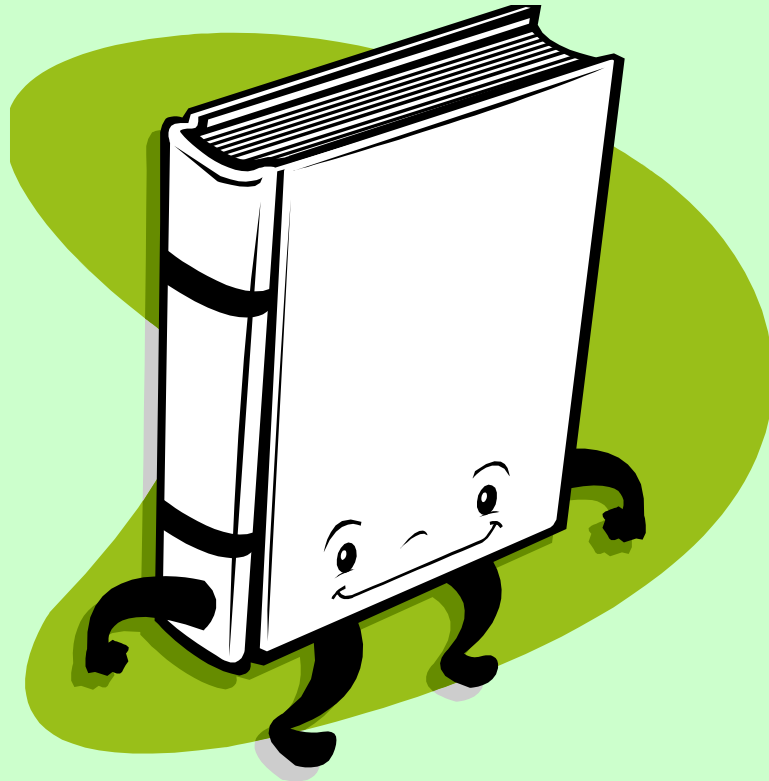
IS NOT NULL, NOT (... IS NULL)

beget, propagate

NVL

NULLS FIRST, NULLS LAST

IFNULL()



Please drop me an email if you noticed any errors in this module. I'd also appreciate reading your comments, criticisms, and or suggestions as to how this module could be improved.

Thanks,

bil



That's All