Tipo Boolean in Oracle – Una piccola digressione

Due parole sul tipo Boolean

In questo post andremo ad affrontare un argomento particolare: parleremo di questo tipo Boolean sotto Oracle.

Esiste il tipo Boolean?

Oracle mette a disposizione il tipo Boolean, ma occorre tenere conto di queste precisazioni:

- NON E' usabile nella definizione dei campi della tabella.
- PUO' essere usato nel PLSQL.

Se vogliamo utilizzarlo nella definizione di una tabella, come suggerito da Tom Kyle, conviene sostituirlo con il tipo ***CHAR(1)***, che presenta le stesse caratteristiche, come mostrato di seguito:

```
flag char(1) check (flag in ( 'Y', 'N' )),
```

Il risultato è il medesimo.

Una precisazione

Anche se esiste come ***datatype***, non è possibile usarlo ovunque. ***PL-SQL ha le sue regole*** e chi lo usa deve mettersi in testa che si devono seguire. Non si può pretendere che, se altri linguaggi (quali ad es. JAVA) lo consentono, PL-SQL si deve adeguare.

Conclusione

Abbiamo visto un piccolo tassello, ma di grande importanza. Questo ci ricorda che ogni linguaggio ha le sue regole e i suoi ambiti di applicazione. Dobbiamo tenerne presente sempre.

Esiste una funzione isNumeric in PLSQL??

Potrebbe sorgere la necessità di dover leggere dei valori solo numerici, memorizzati su di un campo alfanumerico. I motivi per cui si può avere questa necessità possono essere svariati.

Al momento in cui scrivo, una funzione ***isNumeric*** o similare non esiste in PLSQL. Come possiamo realizzare questo? Ci viene in aiuto il sito di AskTom, che ci fornisce alcuni spunti.

Abbiamo due strade:

Funzione Translate….

La funzione Translate può risultare molto utile. Come indicato in questa pagina, la funzione esegue una sostituzione di caratteri basandosi su delle corrispondenze. Di conseguenza, un sistema per ottenere il risultato desiderato, si può usare la seguente istruzione:

replace(translate(<valore>, '0123456789', '0000000000'), '0', '')

Spieghiamo il funzionamento: La translate, fondamentalmente, esegue una sostituzione delle cifre numeriche con '0' (zero). La seconda funzione replace, banalmente, elimina lo '0' (zero).

Conseguenza? Se ***<valore>*** contiene delle cifre numeriche, queste sono eliminate e viene restituita una stringa vuota. In alternativa, se su ***<valore>*** è presente un carattere alfanumerico, questo rimane e viene restituito nella stringa.

.

REGEX – Espressioni Regolari

E' possibile anche utilizzare le espressioni regolari, per riuscire ad identificare i valori solo numerici. Se la versione di Oracle è la 10g o successive, la funzione ***REGEXP_LIKE*** può essere utilizzata. Di seguito un esempio:

SELECT col1 FROM t1 WHERE REGEXP_LIKE(col1, '[[:digit:]]');

Basta semplicemente impostare le regole, ed il gioco è fatto.

Conclusioni

Si tratta di due soluzioni più che valide, ma la prima risulta essere molto semplice, veloce, pratica ed usabile in vari contesti. La seconda risulta più pesante.

Scrivere file con Oracle

Oggi tratteremo un argomento molto semplice. Si tratta della gestione dei file da parte di Oracle.

Oracle consente di poter leggere/scrivere dei file attraverso un package standard: **UTL_FILE;** che mette a disposizione tutti i metodi necessari per poter eseguire qualsiasi operazione sui file.

> ***Una piccola raccomandazione: Prima di poter utilizzare il package, assicurarsi che l'utenza funzionale, cui impostare la stored procedure/package di utilizzo dei file, disponga delle grant di execute per poter usare il package. Se non dispone di tali grant, chiedere all'amministratore del Database di fornirle.***

Come prima cosa, occorre indicare ad Oracle dove si intende andare ad eseguire le operazioni di lettura/scrittura dei file. Per far ciò, occorre definire una DIRECTORY, ovvero indicare ad Oracle dove eseguire le varie operazioni. Per far ciò, si utilizza il seguente comando:

> ***CREATE OR REPLACE DIRECTORY TMP_DIR AS '<path completo della***

```
directory>';
```

Attraverso questa istruzione si indica la directory (locale al server dove è installato il DB Oracle), dove sarà possibile eseguire le varie operazioni di lettura/scrittura dei file. E' bene ribadire che le operazioni di lettura/scrittura sono eseguite sulla macchina locale e non su dei server remoti. Se si ha la necessità di poter scrivere su dei server remoti, occorre disporre di altre procedure, che magari sfruttando il protocollo FTP, possano scrivere i file su server remoti. Package e procedure sono disponibili e saranno trattati in un altro post.

> ***Si segnala altresì che la directory non appartiene ad uno schema specifico. Occorre quindi prestare attenzione al nome della directory da usare, in quanto è univoco per l'intero database.***

Una volta che a directory è stata creata, adesso usiamo i metodi del package per scrivere un semplice file:

```
/*
* Codice di esempio di scrittura di un file
*/

DECLARE

lv_nome_file VARCHAR2(2000);
lv_percorso_file VARCHAR2(2000);
lv_file_id UTL_FILE.file_type;

BEGIN

lv_percorso_file := '<directory_definita>';
lv_nome_file := '<nome_file_da_scrivere>;

– Apertura del file
lv_file_id := UTL_FILE.fopen( lv_percorso_file, lv_nome_file,
```

```
'w' );

-- Scrittura della riga nel file
UTL_FILE.put_line(file => lv_file_id , buffer => 'Questo è un
esempio di testo scritto su di un file' );

-- Chiusura del file.
UTL_FILE.fclose(file => lv_file_id );

EXCEPTION
WHEN OTHERS THEN
DBMS_OUTPUT.PUT_LINE('Errore - ' || TO_CHAR(SQLCODE) || ' - '
|| SUBSTR(SQLERRM,1,150));
END;
```

Analogamente, la lettura di un file viene eseguita con un codice similare e sfruttando gli stessi metodi.

```
/*
* Codice di esempio di lettura di un file
*/

DECLARE
lv_nome_file VARCHAR2(2000);
lv_percorso_file VARCHAR2(2000);
lv_file_id UTL_FILE.file_type;
lv_text VARCHAR2(32767);
BEGIN

lv_percorso_file := '<directory_definita>';
lv_nome_file := '<nome_file_da_scrivere>;;

lv_file_id   :=   UTL_FILE.fopen(   lv_percorso_file,
 lv_nome_file, 'r', 32767);

BEGIN

LOOP

UTL_FILE.get_line(lv_file_id, lv_text, 32767);
```

```
DBMS_OUTPUT.put_line('Line: |' || lv_text || '|');
END LOOP;
EXCEPTION
WHEN NO_DATA_FOUND THEN
NULL;
END;

DBMS_OUTPUT.put_line('Line : |' || lv_text || '|');

UTL_FILE.fclose(lv_file_id);
END;
```

Come si vede il package è molto semplice e consente di poter eseguire tutte le operazioni di cui si abbisogna senza grandi difficoltà.

URL di riferimento

- Semplice articolo in inglese su UTL_FILE
- Indicazioni sulle Directory di Oracle sono presenti su questo link e su questa URL.