]>

USERENV

Description

USERENV built-in function returns information about the current session (CLIENT_INFO, ENTRYID, INSTANCE, ISDBA, LANG, LANGUAGE, SESSIONID, TERMINAL). However, it is a legacy function that has been replaced by SYS_CONTEXT. Example:

```
SELECT USERENV('SESSIONID') FROM dual
```

Solution

See the workaround for SYS_CONTEXT.

Can be converted automatically: NO

TO_DSINTERVAL

Description

The TO_DSINTERVAL function in Oracle's PL/SQL converts a given string to an INTERVAL DAY TO SECOND data type. DB2 does not support the INTERVAL data type, therefore none of the related functions will be supported.

Solution

As possible work around, DB2 supports labeled durations: time_in + seconds_in SECONDS

Note: See "Interval processing" topic (below) for a potential workaround

Can be converted automatically: NO

Interval processing (NUMTODSINTERVAL, NUMTOYMINTERVAL, TO_DSINTERVAL, TO_YMINTERVAL)

Description

There are several Oracle built in functions to assist with converting values into into the INTERVAL_DAY_TO_SECOND and INTERVAL_YEAR_TO_MONTH literals.

Solution

The following set of types and functions should allow you to mimic the required functionality:

```
CREATE TYPE INTERVAL_DAY_TO_SECOND AS DECFLOAT WITH COMPARISONS
@

CREATE TYPE INTERVAL_YEAR_TO_MONTH AS DECFLOAT WITH COMPARISONS
@

CREATE OR REPLACE FUNCTION NUMTODSINTERVAL(val DECFLOAT, unit VARCHAR(10))
RETURNS INTERVAL_DAY_TO_SECOND
RETURN INTERVAL_DAY_TO_SECOND(TRUNC((CASE UPPER(unit)
WHEN 'SECOND' THEN val
WHEN 'MINUTE' THEN val * 60
WHEN 'HOUR' THEN val * 60 * 60
WHEN 'DAY' THEN val * 60 * 60 * 24
ELSE RAISE_ERROR('78000', 'Unknown unit ' || unit || ' for NUMTODSINTERVAL')
END), 6))
@

CREATE OR REPLACE FUNCTION NUMTOYMINTERVAL(val DECFLOAT, unit VARCHAR(10))
RETURNS INTERVAL_YEAR_TO_MONTH
RETURN INTERVAL_YEAR_TO_MONTH(TRUNC(CASE UPPER(unit)
WHEN 'MONTH' THEN val
WHEN 'YEAR' THEN val * 12
ELSE RAISE_ERROR('78000', 'Unknown unit ' || unit || ' for NUMTOYMINTERVAL')
END))
@

CREATE OR REPLACE FUNCTION TO_DSINTERVAL(val VARCHAR(20))
RETURNS INTERVAL_DAY_TO_SECOND
RETURN INTERVAL_DAY_TO_SECOND(
```

```
CASE WHEN LOCATE_IN_STRING(val, ' ') = 0
THEN DECFLOAT(val) * 60 * 60 * 24
ELSE DECFLOAT(SUBSTR(val, 1, LOCATE_IN_STRING(val, ' ') - 1)) * 60 * 60 * 24
+ (CASE WHEN SUBSTR(val, 1, 1) = '-' THEN -1 ELSE 1 END
* CASE WHEN LOCATE_IN_STRING(val, '.') > 0
THEN MIDNIGHT_SECONDS(TIME(SUBSTR(val,
LOCATE_IN_STRING(val, ' ') + 1,
LOCATE_IN_STRING(val, '.') - LOCATE_IN_STRING(val, ' ') - 1)))
+ DECFLOAT(RPAD(SUBSTR(val,
LOCATE_IN_STRING(val, '.') + 1,
LENGTH(val) - LOCATE_IN_STRING(val, '.')),
6, '0')) * 0.000001
ELSE MIDNIGHT_SECONDS(TIME(SUBSTR(val, LOCATE_IN_STRING(val, ' ') + 1,
LENGTH(val) - LOCATE_IN_STRING(val, ' '))))
END)
END)
@

CREATE OR REPLACE FUNCTION TO_YMINTERVAL(val VARCHAR(20))
RETURNS INTERVAL_YEAR_TO_MONTH
RETURN INTERVAL_YEAR_TO_MONTH(
CASE WHEN SUBSTR(val, 1, 1) = '-'
THEN CASE WHEN LOCATE_IN_STRING(val, '-', 2) > 0
THEN DECFLOAT(SUBSTR(val, 1, LOCATE_IN_STRING(val, '-', 2) - 1)) * 12
- DECFLOAT(SUBSTR(val, LOCATE_IN_STRING(val, '-', 2) + 1, LENGTH(val) - LOCATE_IN_STRING(val, '-', 2)))
ELSE DECFLOAT(val) * 12 END
ELSE CASE WHEN LOCATE_IN_STRING(val, '-') > 0
THEN DECFLOAT(SUBSTR(val, 1, LOCATE_IN_STRING(val, '-') - 1)) * 12
+ DECFLOAT(SUBSTR(val, LOCATE_IN_STRING(val, '-') + 1, LENGTH(val) - LOCATE_IN_STRING(val, '-', 2)))
ELSE DECFLOAT(val) END
END)
@
CREATE OR REPLACE FUNCTION TO_CHAR(val INTERVAL_DAY_TO_SECOND)
RETURNS VARCHAR(26)
RETURN TO_CHAR(TRUNC(DECFLOAT(val) / (60 * 60 * 24)), 'S000000000')
|| TO_CHAR(MOD(TRUNC(ABS(DECFLOAT(val)) / (60 * 60)), 24), '00') || ':'
|| SUBSTR(TO_CHAR(MOD(TRUNC(ABS(DECFLOAT(val)) / 60), 60), '00'), 2) || ':'
|| SUBSTR(TO_CHAR(MOD(TRUNC(ABS(DECFLOAT(val))), 60), '00'), 2)
|| SUBSTR(TO_CHAR(TRUNC(ABS(DECFLOAT(val)), 6) - TRUNC(ABS(DECFLOAT(val)))), 2)
@

CREATE OR REPLACE FUNCTION TO_CHAR(val INTERVAL_YEAR_TO_MONTH)
RETURNS VARCHAR(13)
RETURN TO_CHAR(TRUNC(DECFLOAT(val) / 12), 'S000000000') || '-'
|| SUBSTR(TO_CHAR(MOD(TRUNC(ABS(DECFLOAT(val))), 12), '00'), 2)
@

CREATE OR REPLACE FUNCTION "+"(INTERVAL_YEAR_TO_MONTH, INTERVAL_YEAR_TO_MONTH)
RETURNS INTERVAL_YEAR_TO_MONTH
SOURCE SYSIBM."+"(DECFLOAT, DECFLOAT)
@

CREATE OR REPLACE FUNCTION "-"(INTERVAL_YEAR_TO_MONTH, INTERVAL_YEAR_TO_MONTH)
RETURNS INTERVAL_YEAR_TO_MONTH
SOURCE SYSIBM."-"(DECFLOAT, DECFLOAT)
@

-- Put DATE before TIMESTAMP. In ORA mode the later will override DATE
CREATE OR REPLACE FUNCTION "+"(arg1 DATE, arg2 INTERVAL_YEAR_TO_MONTH)
RETURNS DATE
NO EXTERNAL ACTION CONTAINS SQL DETERMINISTIC
RETURN arg1 + DECFLOAT(arg2) MONTHS
@

CREATE OR REPLACE FUNCTION "-"(arg1 DATE, arg2 INTERVAL_YEAR_TO_MONTH)
RETURNS DATE
NO EXTERNAL ACTION CONTAINS SQL DETERMINISTIC
RETURN arg1 - DECFLOAT(arg2) MONTHS
@

CREATE OR REPLACE FUNCTION "+"(arg1 TIMESTAMP(12), arg2 INTERVAL_YEAR_TO_MONTH)
RETURNS TIMESTAMP(12)
NO EXTERNAL ACTION CONTAINS SQL DETERMINISTIC
RETURN arg1 + DECFLOAT(arg2) MONTHS
@

CREATE OR REPLACE FUNCTION "-"(arg1 TIMESTAMP(12), arg2 INTERVAL_YEAR_TO_MONTH)
RETURNS TIMESTAMP(12)
NO EXTERNAL ACTION CONTAINS SQL DETERMINISTIC
RETURN arg1 - DECFLOAT(arg2) MONTHS
@

CREATE OR REPLACE FUNCTION "+"(arg1 INTERVAL_YEAR_TO_MONTH, arg2 DATE)
RETURNS DATE
NO EXTERNAL ACTION CONTAINS SQL DETERMINISTIC
RETURN arg2 + DECFLOAT(arg1) MONTHS
@

CREATE OR REPLACE FUNCTION "+"(arg1 INTERVAL_YEAR_TO_MONTH, arg2 TIMESTAMP(12))
RETURNS TIMESTAMP(12)
NO EXTERNAL ACTION CONTAINS SQL DETERMINISTIC
RETURN arg2 + DECFLOAT(arg1) MONTHS
@
```

```
CREATE OR REPLACE FUNCTION "+"(INTERVAL_DAY_TO_SECOND, INTERVAL_DAY_TO_SECOND)
RETURNS INTERVAL_DAY_TO_SECOND
SOURCE SYSIBM."+"(DECFLOAT, DECFLOAT)
@

CREATE OR REPLACE FUNCTION "-"(INTERVAL_DAY_TO_SECOND, INTERVAL_DAY_TO_SECOND)
RETURNS INTERVAL_DAY_TO_SECOND
SOURCE SYSIBM."-"(DECFLOAT, DECFLOAT)
@

CREATE OR REPLACE FUNCTION "+"(arg1 TIMESTAMP(12), arg2 INTERVAL_DAY_TO_SECOND)
RETURNS TIMESTAMP(12)
NO EXTERNAL ACTION CONTAINS SQL DETERMINISTIC
RETURN arg1 + DECFLOAT(arg2) SECONDS
@

CREATE OR REPLACE FUNCTION "-"(arg1 TIMESTAMP(12), arg2 INTERVAL_DAY_TO_SECOND)
RETURNS TIMESTAMP(12)
NO EXTERNAL ACTION CONTAINS SQL DETERMINISTIC
RETURN arg1 - DECFLOAT(arg2) SECONDS
@

CREATE OR REPLACE FUNCTION "+"(arg1 INTERVAL_DAY_TO_SECOND, arg2 TIMESTAMP(12))
RETURNS TIMESTAMP(12)
NO EXTERNAL ACTION CONTAINS SQL DETERMINISTIC
RETURN arg2 + DECFLOAT(arg1) SECONDS
@
```

Can be converted automatically: NO