DML Features - Reverse

This function reverses the string input …

reverse('ABC')

is output as

CBA

DML Features - First/Last_Value

These two functions will return either the first value in a column in all rows or the last value in a column, this works rather like using a 'string' within a query as it populates all rows with the same value.

The syntax for both is as follows …

first_value(<column_name>) over (order by <column_name>)

To alias the result use the following …

as <column_alias>

DML Features - First/Last_Value

```
select si_stock_number No
, first_value(si_stock_description) over (order by si_stock_description) as Description
from stock_information;
```

NO	DESCRIPTION
11	BATH MAT SET
3	BATH MAT SET
22	BATH MAT SET
9	BATH MAT SET

DML Features - Lead and Lag

Lead allows two records to be compared without the need for complex self joins or correlated sub-queries, this is a particularly useful function when attempting to find ranges within a table, for example dates or prices …

The syntax for Lead is as follows …

lead(<column_name>, <number of records>) over (order by <column_name>)

The number of records is the number of records from the current to be produced, this represents records succeeding the current, use Lag to use records preceding the current

DML Features - Lead and Lag

This example shows the prices of stock and the differences between the next price using lead …

```
select to_char(si_unit_price, '999990.99') "Unit Price"
,to_char(lead(si_unit_price, 1) over (order by si_unit_price), '999990.99') as "Next Price"
,to_char(((lead(si_unit_price, 1) over (order by si_unit_price)) - si_unit_price), '999990.99')
    "Difference"
from stock_information
order by si_unit_price;
```

Unit Price	Next Price	Difference
15.99	16.99	1.00
16.99	29.99	13.00
29.99	79.99	50.00

DML Features - Trim

Trim is effectively an extension of the Rtrim and Ltrim functions, in that it can be used to trim both ends of a string simultaneously, or can be used to trim a specified edge..
The syntax is a little more complicated than standard functions …

trim([leading/trailing/both] <value> from <value>)

Leading and trailing specify the direction of the trim, leaving either them or both out will default to both … therefore

trim('a' from 'aaaaaakkkkkaaaaa') will result in ..
kkkkk

DML Features - Trim

trim(leading 'a' from 'aaaaaakkkkkaaaaa') will result in .. *kkkkkaaaaa*

And *trim(trailing 'a' from 'aaaaaakkkkkaaaaa')* will result in .. *aaaaaakkkkk*

In addition, the values need not be strings, it is perfectly legal to specify the removal of one column from another …providing the first column is one character only or uses substr

trim(<column_name> from <column_name2>) from <table_name>