Typically the hierarchy will be defined in a table by the use of a foreign key or 'pigs ear', where a column perhaps called Parent ID would have a foreign key to the Primary Key column.

ID	Level	Name	Parent ID
1	One	Level 1	
3	Two	Level 2 A	1
5	Two	Level 2 B	1

Tree walking is the technique of taking the Parent ID and traversing either up or down the hierarchy or 'tree'.

select <column_name>
from <table_name>
***start with** <column_name> = '<value>'*
***connect by prior** <primary key column> = <foreign key name>;*

In the example this is how to work down the tree….

***start with** name = 'LEVEL 1'*
***connect by prior** ID = Parent_ID;*

To work up the tree the 'prior' clause is moved to the right …

***start with** name = 'LEVEL 6A'*
*connect by ID = **prior** Parent_ID;*

Issuing the following statement on this data will return the following records...

```
select id, level, name, parent_id
from table
start with name = 'LEVEL 1'
connect by prior id = parent_id;
```

ID	Level	Name	Parent_ID
1	One	Level 1	
3	Two	Level 2A	1
5	Two	Level 2B	1

So far Tree Walks have been output in no particular order within the levels, 'order siblings' will allow the ordering of each sub-grouping amongst itself

```
select lpad('-', level,'-')||sl_location_name LOCATION
from store_locations
start with sl_parent_location_id is null
connect by prior sl_location_id = sl_parent_location_id
order siblings by sl_location_name;
```

Note the use of level, this is a reserved word which holds the level value of the current record

Tree Walks - Advanced Ordering of Siblings in Tree Walks

```
LOCATION
---------------
-WEST NORWOOD
--GLASGOW
---EDINBURGH
----DUNDEE
----DUNFERMLINE
----KIRKCALDY
----MIDLOTHIAN
----PAISLEY
----PENRITH
```