Some special Oracle features.

DUAL table

```
SELECT SYSDATE FROM DUAL;
SELECT USER    FROM DUAL;
```

GROUPING functions

You can nest them in depth 1

```
SELECT MAX(AVG(sal)) FROM emp GROUP BY deptno;
```

Top-N analysis

From Oracle version 12 you can use the following syntax:

```
SELECT ename, sal FROM emp ORDER BY sal FETCH FIRST 4 ROWS ONLY;
```

The detailed syntax you can find in the documentation:

```
[ OFFSET offset { ROW | ROWS } ]
[ FETCH { FIRST | NEXT } [ { rowcount | percent PERCENT } ]
    { ROW | ROWS } { ONLY | WITH TIES } ]

SELECT sal FROM emp ORDER BY sal FETCH FIRST 4 ROWS WITH TIES;
```

UNIQUE

Synonym of DISTINCT

Outer join

In Oracle you can use a special syntax too for outer join: (+)

Example:

We have two tables, A(o1, o2) and B(o1,o2) with the following tuples.

A

2	B
4	D

B

3	C
4	D

```
create table a(o1 integer, o2 varchar2(20));
create table b(o1 integer, o2 varchar2(20));
insert into a values(2, 'B');
insert into a values(4, 'D');
insert into b values(3, 'C');
insert into b values(4, 'D');
```

2 equivalent queries:

```
SELECT * FROM A LEFT OUTER JOIN B ON A.O1 = B.O1;
SELECT * FROM A, B WHERE A.O1 = B.O1 (+);
```

Result:

4	D	4	D
2	B	**NULL**	**NULL**

2 modified queries:

```
SELECT * FROM A LEFT OUTER JOIN B ON (A.O1 = B.O1 AND B.O1 > 5);
SELECT * FROM A, B WHERE A.O1 = B.O1 (+) AND B.O1 (+) > 5;
```

Result:

4	D	**NULL**	**NULL**
2	B	**NULL**	**NULL**

2 other equivalent queries:

```
SELECT * FROM A LEFT OUTER JOIN B ON A.O1 = B.O1 WHERE B.O1 > 5;
SELECT * FROM A, B WHERE A.O1 = B.O1 (+) AND B.O1 > 5;
```

Result:

```
No rows selected
```

The examples above point out that the join condition and selection condition (WHERE) are different in case of outer joins. If a tuple doesn't satisfy the join condition, it will not be dropped from the resultset. However, if it doesn't satisfy the selection condition, it will be dropped.