*** Create a table and populate it with 1 million IDs
*** Note it uses a non-ASSM LMT tablespace with 8K block size

SQL> create table test1 as select rownum id from dual connect by level <= 1000000;
Table created.

*** Create a non unique index with a pctfree of 0 in order to pack as many row entries into a leaf block as possible

SQL> create index non_unique_idx on test1(id) pctfree 0;
Index created.

*** A dump of the block shows that the index indeed is non-unique as the rowid constitutes a second column for the index row entry
*** The length of the index row entry is 14 bytes and we can fit 500 (count starts at 0) index row entries in our 8K block

*** Partial Block Dump ...

Leaf block dump
===============
header address 143336028=0x88b225c
kdxcolev 0
KDXCOLEV Flags = - - -
kdxcolok 0
kdxcoopc 0x80: opcode=0: iot flags=--- is converted=Y
kdxconco 2
kdxcosdc 0
kdxconro 500
kdxcofbo 1036=0x40c
kdxcofeo 1042=0x412
kdxcoavs 6
kdxlespl 0
kdxlende 0
kdxlenxt 75520140=0x480588c
kdxleprv 75520138=0x480588a
kdxledsz 0
kdxlebksz 8036
row#0[8022] flag: ------, lock: 0, len=14  <=== length is 14 bytes for the index row entry
col 0; len 4; (4):  c3 60 61 1c
col 1; len 6; (6):  04 80 50 3c 01 06  <=== rowid is stored as a second column for the index row entry
row#1[8008] flag: ------, lock: 0, len=14
col 0; len 4; (4):  c3 60 61 1d
col 1; len 6; (6):  04 80 50 3c 01 07
...

row#499[1042] flag: ------, lock: 0, len=14  <=== can fit 500 row entries in the index leaf block
col 0; len 4; (4):  c3 61 02 1b
col 1; len 6; (6):  04 80 50 3d 00 67
*** Now create an identical table, with the same number of rows
*** Note it also uses a non-ASSM LMT tablespace with 8K block size

SQL> create table test2 as select rownum id from dual connect by level <= 1000000;
Table created.

*** This time create a unique index instead

SQL> create unique index unique_idx on test2(id) pctfree 0;
Index created.

*** A dump of the block shows that the index indeed is unique as the rowid
does not constitute a second column for the index row entry
*** The length of the index row entry is therefore reduced by the 1 column
length byte to 13 bytes
*** and we can fit now fit 533 index row entries in our 8K block

Leaf block dump
==================
header address 143336020=0x88b225c
kdxcolelev 0
KDXCOLEV Flags = - - -
kdxcolok 0
kdxcoopc 0x80: opcode=0: iot flags=- - - is converted=Y
kdxconco 1
kdxcosdc 0
kdxconro 533
kdxcofbo 1102=0x44e
kdxcofeo 1112=0x458
kdxcoavs 10
kdxelepl 0
kdendlene 0
kdxbxnt 75527436=0x480750c
kdxbxprv 75527434=0x480750a
kdxledsz 6
kdxlebksz 8036
row#0[8023] flag: ------, lock: 0, len=13, data:(6): 04 80 5e 34 02 82
<== length is 13 bytes and rowid not stored as a second column entry
col 0; len 4; (4): c3 60 30 2c
row#1[8010] flag: ------, lock: 0, len=13, data:(6): 04 80 5e 34 02 83
col 0; len 4; (4): c3 60 30 2d
...
row#532[1112] flag: ------, lock: 0, len=13, data:(6): 04 80 5e 35 02 04
<== can fit 533 index row entries in the block
col 0; len 4; (4): c3 60 35 4c
----- end of leaf block dump -----

----- end of leaf block dump -----
*** If we look at the number of leaf blocks required for each index ...

SQL> exec dbms_stats.gather_index_stats(ownname=>'BOWIE',
indname=>'NON_UNIQUE_IDX', estimate_percent=> null);

PL/SQL procedure successfully completed.

SQL> exec dbms_stats.gather_index_stats(ownname=>'BOWIE',
indname=>'UNIQUE_IDX', estimate_percent=> null);

PL/SQL procedure successfully completed.

SQL> select index_name, blevel, leaf_blocks, num_rows from dba_indexes
where index_name IN ('NON_UNIQUE_IDX', 'UNIQUE_IDX');

<table>
<thead>
<tr>
<th>INDEX_NAME</th>
<th>BLEVEL</th>
<th>LEAF_BLOCKS</th>
<th>NUM_ROWS</th>
</tr>
</thead>
<tbody>
<tr>
<td>NON_UNIQUE_IDX</td>
<td>2</td>
<td>1999</td>
<td>1000000</td>
</tr>
<tr>
<td>UNIQUE_IDX</td>
<td>2</td>
<td>1875</td>
<td>1000000</td>
</tr>
</tbody>
</table>

*** Note the Non-Unique index uses approximately 6.6% more leaf blocks to store the same amount of IDs