Database Systems II B: DBMS-Implementation

Chapter 8: Row Format

Prof. Dr. Stefan Brass

Martin-Luther-Universität Halle-Wittenberg

Wintersemester 2019/20

http://www.informatik.uni-halle.de/~brass/dbi19/

Objectives

After completing this chapter, you should be able to:

- write a short paragraph explaining how blocks are allocated in Oracle (mention segments, extents).
- find storage information in the data dictionary.

  And use the `ANALYZE TABLE` command to populate the dictionary tables.
- explain how relations are stored in Oracle (row and block format, TIDs/ROWIDs, migrated rows).
- estimate the number of blocks needed for a table.
- set the basic storage parameters for relations in Oracle for good performance.

Inhalt

Data Dictionary: COLS

- COLS (a synonym for USER_TAB_COLUMS) contains the following information set by the ANALYZE TABLE:
  - TABLE_NAME, COLUMN_NAME: Identifies the column.
  - NUM_DISTINCT: Number of distinct data values.
  - NUM_NULLS: Number of rows for which this column is null.
  - LOW_VALUE, HIGH_VALUE: Smallest/greatest value.

    They are shown in the internal format (not readable).
- See also USER_TAB_COL_STATISTICS.

Row Format (1)

- Normal row format in Oracle (not chained, not clustered):

Row Header	1st Col. Length	1st Col. Data	2nd Col. Length	2nd Col. Data	...

- The row header contains the number of columns and the number of chain pieces (3 bytes in total).
- The column length is encoded in one byte if below 250. Otherwise it needs three bytes.

Row Format (2)

- The length of the column data depends on the data type. E.g. a `VARCHAR`-string with 5 characters needs 5 byte.

  See below for more information.

- In the order of columns is normally the order of declaration in the `CREATE TABLE` statement.

  But `LONG` columns are moved towards the end. Columns added with `ALTER TABLE` are also added at the end.

- Null values need only the length byte (0).

  If the columns at the end are all filled with null values, they are not stored at all.

Row Format (3)

- The Oracle Row Format is quite compact.
- However, if Oracle wants to access e.g. the fifth attribute, it needs to look at each of the preceeding column lengths.
- Normally the bottleneck is disk I/O, but not the CPU.

  An additional advantage is that when Oracle has to work with these data elements, e.g. strings, it can simply pass a pointer to the length around. So one can see the format also as the concatenation of data values, which encode their own length.
- Exercise: Discuss alternative row formats.

Inhalt

Data Formats (1)

- The storage size of any data value can be determined with the function VSIZE:

```
SELECT SSN, VSIZE(SSN), LNAME, VSIZE(LNAME)
FROM   STUDENT
```

- This is also possible without storing the value:

```
SELECT VSIZE(-1.2), VSIZE('abc')
FROM   DUAL
```

- To see the internal representation of e.g. 123, use

```
SELECT DUMP(123, 16) FROM DUAL
```

The bytes are printed in hexadecimal notation (selected with the argument 16).

This also works with other data types, e.g. `DUMP('ab',16)`.

Data Formats (2)

- `CHAR(`*n*`)`: A fixed-length string is stored in *n* Bytes (one character per byte, filled with blanks to the length *n*).
- `VARCHAR(`*n*`)`: Here only the actual characters are stored. (If a `VARCHAR(10)` column contains `'Jim'`, it needs 3 Byte.)
- The strings are stored in the DB character set (e.g. ASCII).

  The character set can be chosen at DB creation time, but until Oracle 8i it could not be changed later. Now it can be changed to supersets that have the same codepoint values for the subset. Clients can use a different character set and Oracle does the conversion. Oracle can manage multi-byte character sets. Oracle has also types `NCHAR`/`NVARCHAR` for storing strings in a second, national character set.

Data Formats (3)

- `RAW(n)`: Variable-length string of data which is not interpreted / not converted between different character sets.

  Input/output is in form of a string of hexadecimal digits.

  `RAW(10)` means max. 10 Byte, but `'00FF'` needs 2 Byte.

- `BLOB`, `CLOB`: Large Objects.

  One can choose `ENABLE STORAGE IN ROW` (the default) or `DISABLE STORAGE IN ROW`. If disabled, 20 bytes are needed in the row for the LOB locator (an ID permitting to access the actual data). If enabled, data up to 3964 bytes are stored within the row. This needs 36 bytes plus the actual data (in total at most 4000 bytes). If the data is longer, it is moved out of the row, but the addresses of the first 12 storage chunks are stored in the row, which needs up to 84 bytes (4 byte per chunk, e.g. $36 + 4 = 40$ byte if the data fits in one chunk). If more than 12 chunks are needed, a LOB index is used. [LOB performance guidelines].

Data Formats (4)

- `NUMBER(p)`, `NUMBER(p,s)`: Numbers are stored in scientific notation with mantissa and exponent.
  E.g. $123 = 1.23 * 10^2$.

  It seems that Oracle really stores it as $1.23 * 100^1$.

  Note that `NUMBER(p,s)` is an Oracle-specific synonym for `NUMERIC(p,s)`.

- The exponent needs always one byte, the mantissa needs one byte per two digits (leading/training zeros are not stored).

  Even if the column is `NUMBER(30)`, 123 needs only 3 Byte.

Data Formats (5)

- So a positive number with n digits needs

  $$1 + \mathrm{ceil}(n/2)$$

  bytes.

  The Oracle 8 Concepts manual says something different.

- Negative numbers need one more byte for the sign.
- Oracle can store up to 38 significant digits, thus a number needs at most 21 Byte (or 20 Byte if positive).

Data Formats (6)

- `ROWID`: Physical pointer to a row, needs 10 bytes.

  Maybe: Object 4 Byte, File+Block 4 Byte, Row 2 Byte (?).

- `DATE`: Timestamp (Date and Time), needs 7 Byte.

  Year: 2 Byte, Month: 1 Byte, Day: 1 Byte, Hours: 1 Byte, Minutes: 1 Byte, Seconds 1 Byte. In the default format for input/output (`DD-MON-YY`) only the date portion can be specified and Oracle assumes 0:00am (midnight). However, `SYSDATE` returns not only the current date, but also the time.

Experiments/Exercises (1)

- `CREATE TABLE R(A NUMBER(4), B VARCHAR(50))`.
- `INSERT INTO R VALUES (12, 'abcde')`.
- What will be the output of this query?

```
SELECT A, VSIZE(A), B, VSIZE(B) FROM R;
```

- `ANALYZE TABLE R COMPUTE STATISTICS`.
- What will be the output of this query?

```
SELECT AVG_ROW_LEN FROM TABS
WHERE  TABLE_NAME = 'R';
```

Experiments/Exercises (2)

- TABS also reports
  - INITIAL_EXTENT=10240,
  - BLOCKS=1,
  - NUM_ROWS=1,
  - EMPTY_BLOCKS=3,
  - AVG_SPACE=1944,
  - NUM_FREELIST_BLOCKS=1.

  Please explain (blocksize is 2048).

Experiments/Exercises (3)

- When another row is inserted, e.g. `(34,'uvwxy')`, the `AVG_SPACE` shrinks to 1930. Please explain.
- When this row is deleted again, `AVG_SPACE` grows only to 1942. Why?
- The procedure on the next slide is used to insert rows of the above length until the first block is full. 125 rows are inserted before the system starts to use a second block.

  This table is declared with `PCTFREE=10`. `TABS` shows e.g.: `BLOCKS=2`, `NUM_ROWS=126`, `EMPTY_BLOCKS=2`, `AVG_SPACE=1076`, `NUM_FREELIST_BLOCKS=1`, `AVG_SPACE_FREELIST_BLOCKS=1944`. Please explain.

Experiments/Exercises (4)

```
(1)  CREATE OR REPLACE PROCEDURE P AS
(2)      N NUMBER;
(3)  BEGIN
(4)      N := 1;
(5)      WHILE N < 2 LOOP
(6)          INSERT INTO R VALUES(34, 'uvwxy');
(7)          SELECT COUNT(DISTINCT DBMS_ROWID.
(8)                   ROWID_BLOCK_NUMBER(ROWID))
(9)          INTO N FROM R;
(10)     END LOOP;
(11) END;
```

References

- Elmasri/Navathe: Fundamentals of Database Systems, 3rd Edition. Section 5.5,5.7.
- Ramakrishnan/Gehrke: Database Management Systems, 2nd Edition. Section 7.3, 7.5–7.8.
- Silberschatz/Korth/Sudarshan: Database System Concepts, 3rd Ed., Chap 10.
- Kemper/Eickler: Datenbanksysteme (in German), Chap. 7, Oldenbourg, 1997.
- Garcia-Molina/Ullman/Widom: Database System Implementation. Chapter 3.
- Härder/Rahm: Datenbanksysteme, Konzepte und Techniken der Implementierung (in German). Chapter 2, 5.
- Jason S. Couchman: Oracle8i Certified Professional: DBA Certification Exam Guide with CDROM. Osborne/ORACLE Press, ISBN 0-07-213060-1, ca. 1257 pages.
- Mark Gurry, Peter Corrigan: Oracle Performance Tuning, 2nd Edition (with disk).
- Gray/Reuter: Transaction Processing: Concepts and Techniques. 1993.
- Oracle 8i Concepts, Release 2 (8.1.6), Oracle Corporation, 1999, Part No. A76965-01.
- Oracle 8i Designing and Tuning for Performance, Release 2 (8.1.6), Oracle Corporation, 1999, Part No. A76992-01.