DB2 10 Inline LOBS

Sandi Smith
BMC Software Inc.

Session Code: E10
2011, November 16 9:45-10:45 | Platform: DB2 for z/OS

Introduction

- LOBs have been available since DB2 V6
- Need for LOBs because of limitations of VARCHAR
  - Maximum length of VARCHAR is 32K
  - Every row has maximum bytes allocated
- LOBs allowed storage of data in a separate space
- Improvements to LOBs in each release
- INLINE LOBs introduced in DB2 V10 – ENFM mode

Agenda

- Understand how INLINE LOBs work
- Learn how to create INLINE LOBs
- Discuss how to determine the best INLINE LENGTH value.
- Pinpoint the new V10 features that affect INLINE LOBs (ZPARMS,utilities)
- Explore how INLINE LOBs effect processing regarding DASD savings and processing savings.

A short history of LOBs

- V6

  EBCDIC & ASCII

- V7

  UNICODE, ability to use LOAD & UNLOAD

  LOBS materialized using data spaces

  CHECK LOB – identify structural defects in LOB TS and invalid LOB values

  CHECK DATA – validate consistency between base and auxiliary tables

- V8

  LOBs materialized using DBM1 storage above 2GB

  ZPARMS to limit storage allocation for LOB materializaation

  - LOBVALA – size per user
  - LOBVALS – size per system

  Auto genation of ROWID column and base table

  XMB2CLOB function – convert XML value into a CLOB

LOBs in V9

- LOAD/UNLOAD - ability to use file reference
- ZPARM MAXOFILR – control max number of file references concurrently open
- REORG removes embedded free space, attempts to make LOB pages contiguous and uses shadow data sets
- REORG SHRLEVEL REFERENCE – longer availability during reorg
- Avoidance of LOB locking - LOB lock is no longer required to serialize the consistency between the value of the LOB and the column of the base row for an uncommitted read, resulting in fewer locks and less locking overhead
- Automatic creation of objects – omit IN clause in CREATE TABLE
- FETCH CONTINUE – way for applications to read LOB data without having to materialize the entire LOB in memory or use LOB LOCATORs
- CHECK LOB – SHRLEVEL CHANGE
- CHECK DATA – SHRLEVEL CHANGE

LOBs in V10

- REORG enhancements
  - Allow SHRLEVEL CHANGE
  - Permit rows to move between partitions
  - Add AUX YES syntax
- INLINE LOBs introduced
- Eliminate LOB materialization within DB2
- Alter AUX tablespace and index page size
- DEFINE NO on AUX tablespace and index
- Spanned records in LOAD and UNLOAD
- Alter maximum length of LOB column

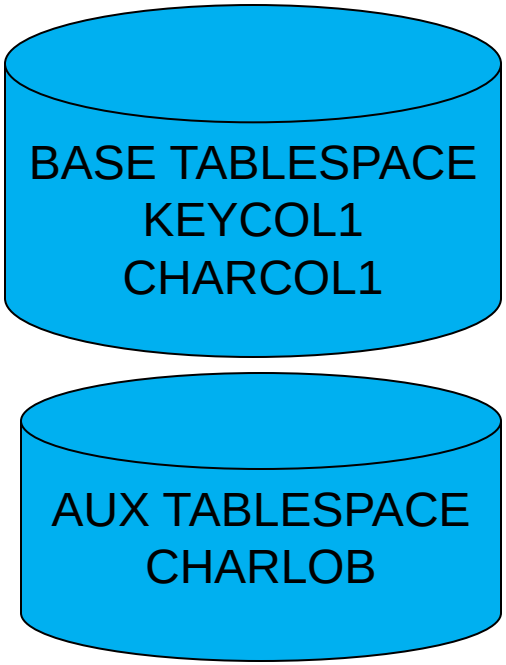
How do INLINE LOBS work?

- The data for an outline LOB column (non-inline LOB) is contained entirely in the auxiliary table
- The data for an INLINE LOB column is contained in the base table and (optionally) in the auxiliary table
- The size of your data and the value of the INLINE LENGTH determine what data is stored where.

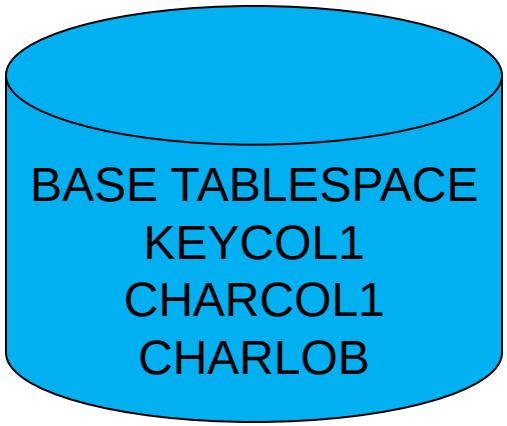
```
CREATE TABLE BMCTABLE
     (KEYCOL1    INTEGER
     ,CHARCOL1   CHAR(20)
     ,CHARLOB    CLOB(1M) INLINE LENGTH( 2000))
```

CHARLOB contains 1024 bytes of data

OUTLINE LOB



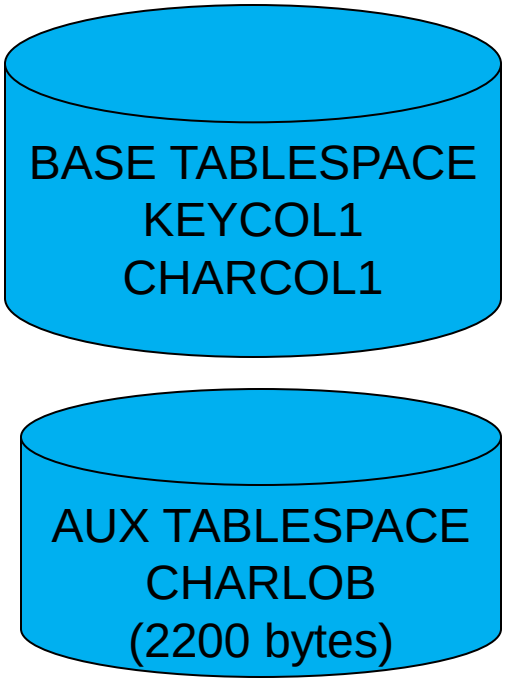
INLINE LOB



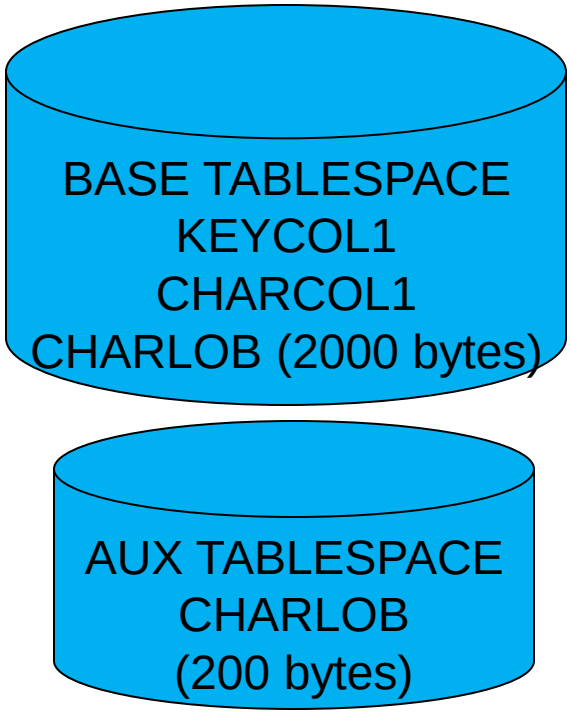
```
CREATE TABLE BMCTABLE
     (KEYCOL1    INTEGER
     ,CHARCOL1   CHAR(20)
     ,CHARLOB    CLOB(1M) INLINE LENGTH( 2000))
```

CHARLOB contains 2200 bytes of data

OUTLINE LOB



INLINE LOB



How do you create an INLINE LOB?

Universal Table Space only

Must be Reordered Row Format

Define No on AUX table space

INLINE LOB columns can be used in index on expression

- SUBSTR only, INLINE portion only
- Start and end values must be constants

If you use TYPE to define INLINE LOB, you cannot use LOB column in index on expression and you cannot specify a default value

How do you create an INLINE LOB?

```
CREATE TYPE BMC.BMC_CHARLOB AS CLOB(1M) CCSID ASCII INLINE
    LENGTH 32680;
```

```
CREATE TABLE BMC.TABLE1
      (KEYCOL1     INTEGER
       ,CHARLOB    BMC.BMC_CHARLOB
```

```
CREATE TABLE BMC.TABLE2
      (KEYCOL1     INTEGER
       ,BINLOB     BLOB(1M)  INLINE LENGTH (32680)
                   WITH DEFAULT   CAST('BMC  IS THE BEST!'AS BLOB)
```

```
CREATE TABLE BMC.TABLE3
      (KEYCOL1     INTEGER
      ,DBCHARLOB     DBCLOB(1M) INLINE LENGTH (16340)
```

```
CREATE INDEX BMC.INDEX1 ON BMC.TABLE1
 (CHAR(SUBSTR(CHARLOB,1,200)) )
  USING STOGROUP AFR10003SG ;

 SELECT DBCHARLOB FROM BMC.TABLE1 WHERE
   CHAR(SUBSTR(CHARLOB,21,10)) = ‘pick me’;
```

What is a good value for INLINE LENGTH?

Factors that affect the decision

- LOB column frequency of reference
- Distribution of LOB column sizes
- Page size of base tablespace
- Compressibility of LOB data
- Search requirements on LOB data

Information and discussion of these topics can be found in DB2 V10 Performance Topics redbook publication

LOB column frequency of reference

- If LOB column is rarely referenced, storing the LOB column inline will increase size of row, resulting in fewer rows stored per page and lower hit ratio

- If LOB column is frequently referenced, storing the LOB column inline will result in fewer I/O and possible DASD savings. But, other factors need to be considered too.

Distribution of LOB column sizes

- Need to consider size of the row and size of the page
- If the LOB column will fit entirely inline 90% of the time, then it is beneficial to inline the LOB
- If the LOB column will overflow 90% of the time, it doesn't make sense to inline the LOB (increased I/O)
- Statistics on the distribution of LOB column lengths are not maintained in the DB2 catalog. DB2 V10 Peformance Topics provides a sample SQL query to print column size distribution for your data (page 109)

SQL query to display LOB column sizes

- Query uses LENGTH function to pull the actual length of the LOB column for each row

- ```
  WITH LOB_DIST_TABLE (LOB_LENGTH, LOB_COUNT) AS(SELECT
  LOBCOL_LENGTH, COUNT(*)
       FROM(SELECT ((LENGTH(lob column name) / 4000) + 1) * 4000
       AS LOBCOL_LENGTH FROM table name) LOB_COL_LENGTH_TABLE GROUP
          BY LOBCOL_LENGTH)
     SELECT '04000', SUM(LOB_COUNT) FROM LOB_DIST_TABLE WHERE
            LOB_LENGTH <= 4000      UNION
  SELECT '08000', SUM(LOB_COUNT) FROM LOB_DIST_TABLE WHERE
            LOB_LENGTH <= 8000      UNION  …..
  ```

- The end result is a report that shows how many LOB rows fit into each multiple of 4000 bytes in length.
  ```

Example of output from SQL query to display LOB column sizes

LOB_LENGTH	LOB_COUNT	
04000	35208	<< of the 35263 rows, 35208 are 4000 or less
…..		
16000	35255	
20000	35263	
24000	35263	<< 0 are between 20001 and 28000
….		
99999	35263	<< 35263 total rows

Counts are a cumulative distribution

Helps you determine what page size and inline length to use

Display number of LOB columns

If you want to find out how many LOBs you have in a specific database or schema, you can issue an SQL statement like this

```
SELECT SUBSTR(TBCREATOR,1,15) AS TBCREATOR,
       SUBSTR(TBNAME,1,15) AS TBNAME,
       SUBSTR(NAME,1,15) AS COLNAME,
       COLTYPE, LENGTH2
       FROM SYSIBM.SYSCOLUMNS
         WHERE TBCREATOR LIKE 'SYS%'
           AND COLTYPE IN ('BLOB','CLOB','DBCLOB')  ;
```

Output from query to display number of LOB columns

TBCREATOR	TBNAME	COLNAME	COLTYPE	LENGTH2
SYSIBM	SYSQUERY_AUX	AUXVALUE	CLOB	209715
SYSIBM	SYSROUTINES	TEXT	CLOB	209715
SYSIBM	SYSROUTINES	PARSETREE	BLOB	107374182
SYSIBM	SYSROUTINESTEXT	AUXVALUE	CLOB	209715
SYSIBM	SYSROUTINES_TRE	AUXVALUE	BLOB	107374182
SYSIBM	SYSTABLES_PROFI	PROFILE_TEXT	CLOB	104857
SYSIBM	SYSTRIGGERS	STATEMENT	CLOB	209715
SYSIBM	SYSTRIGGERS_STM	AUXVALUE	CLOB	209715
SYSIBM	SYSVIEWS	STATEMENT	CLOB	209715
SYSIBM	SYSVIEWS	PARSETREE	BLOB	107374182
SYSIBM	SYSVIEWS_STMT	AUXVALUE	CLOB	209715

Display number of INLINE LOB columns

If you want to find out how many INLINE LOBs you have in a specific database or schema, you can issue an SQL statement like this

```
SELECT SUBSTR(TBCREATOR,1,15) AS TBCREATOR,
       SUBSTR(TBNAME,1,15) AS TBNAME,
       SUBSTR(NAME,1,15) AS COLNAME,
       COLTYPE, LENGTH
       FROM SYSIBM.SYSCOLUMNS
         WHERE TBCREATOR LIKE 'SYS%'
           AND COLTYPE IN ('BLOB','CLOB','DBCLOB')  ;
           AND LENGTH > 4;
```

Output from query to display number of INLINE LOB columns

TBCREATOR	TBNAME	COLNAME	COLTYPE	LENGTH
SYSIBM	SYSCONTR	RULETEXT	CLOB	16004
SYSIBM	SYSCONTR	DESCRIPT	BLOB	12004
SYSIBM	SYSPACKS	STATEMEN	CLOB	15364
SYSIBM	SYSPACKS	STMTBLOB	BLOB	7172
SYSIBM	SYSQUERY	STMTTEXT	CLOB	2052
SYSIBM	SYSVIEWS	PARSETRE	BLOB	27674

Page size of base table space

- Determines how many rows fit on a page. For outline LOBS, only one row can be stored on a LOB tablespace page. With INLINE LOBS, you can fit many on a page.
- Small page size and large lob = few rows per page
- Large page size and small lob = many rows per page
- Alter INLINE LENGTH, alter BUFFERPOOL, and REORG allows experimentation

Compressibility of LOB data

- LOB data in AUX table spaces cannot be compressed
- Compression of INLINE LOB data can save space and I/O by storing multiple rows per page
- CLOBs are good compression candidates
- Use DSN1COMP to gather information about compressibility

LOB column search requirements

- Able to build index on expression on INLINE LOB column
- Restricted to SUBSTR expression
- Allows search for a text string in LOB data

ZPARM that effects INLINE LOB

- LOB_INLINE_LENGTH

  Determines inline length value if none specified in DDL

  Default value is 0 (outline LOB). BMC recommends you do not change this value

  Valid values 0-32680 (bytes)

  If this value is greater than lob length in the DDL, DB2 will use the lob length as the inline length

  If the total length of the table columns plus the inline length exceed a page size, DB2 will issue a -670 error

  -670, ERROR: THE RECORD LENGTH OF THE TABLE EXCEEDS THE PAGE SIZE LIMIT

Alters available for INLINE LOBs

- Alter Add INLINE LENGTH to existing LOB column
  Space left in AREO status after alter
  Cannot create index on expression until after a reorg

- Alter increase existing INLINE LENGTH
  Space left in AREO status after alter
  Newly inserted rows will have new inline length
  Cannot increment to length greater than LOB column length

- Alter decrease existing INLINE LENGTH
  Space left in REORP status after alter
  Cannot decrement if space has index on expression
  Cannot decrement if new inline length is less than default

- Alter add LOB column with INLINE LENGTH to existing table

REORG support for INLINE LOBs

- If a LOB is not fully inlined, base and auxiliary objects should be reorganized in the same step. Use AUX YES and SHRLEVEL REFERENCE.
- With AUX Yes you must use TEMPLATE for copy datasets
- UNLOAD ONLY and UNLOAD EXTERNAL are not available for tables with INLINE LOB

LOAD/UNLOAD support for INLINE LOBs

- UNLOAD/LOAD with spanned records
  Don’t need LOB File Reference Variables

- UNLOAD EXTERNAL is invalid for INLINE LOBs

What are the advantages of using INLINE LOBS? DASD savings.

- Disk space savings:

  Inline LOBS allow more than one LOB per page

  Inline portion of a LOB can be compressed

- If LOB size is 4K and the page size is 4K, then there are no DASD savings because you can only store one LOB in a page of a base table space
- If LOB is 1K and the page size is 4K, then DASD savings is significant. Each inline LOB consumes 1K of DASD, while each LOB stored in a LOB table space consumes 4 K of DASD space, or four times as much DASD
- DASD savings for inline LOBs are greatest when you have small LOBs and many of them. Compression can also contribute to savings for inline LOBs.

What are the advantages of using INLINE LOBS? CPU savings.

- Performance Enhancement

  Synchronous I/Os to the AUX index and LOB table space are avoided for fully INLINE LOB

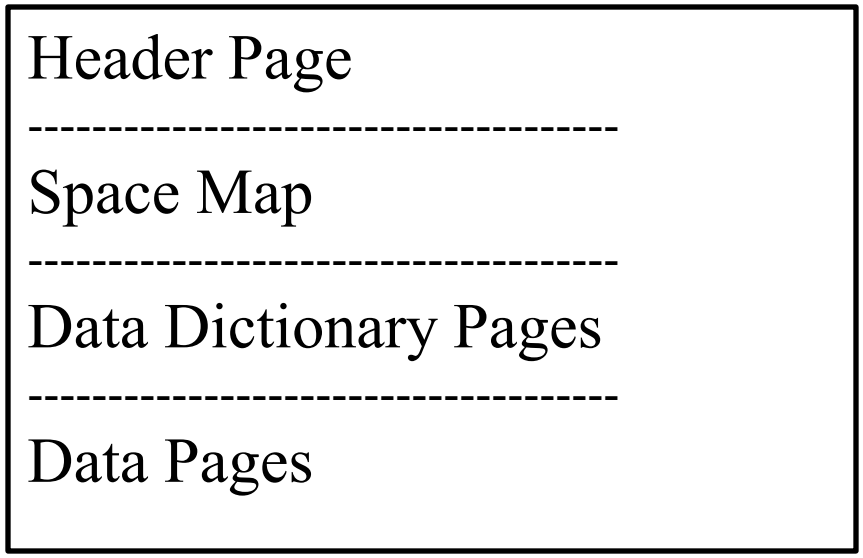
  CPU savings associated with accessing the AUX index and LOB table space for fully INLINE LOB

  Index on expression can be enabled for LOB data to provide more direct access

Base Table Space – Format

- Header
- Space Map
- Data Dictionary Pages
- Data Pages

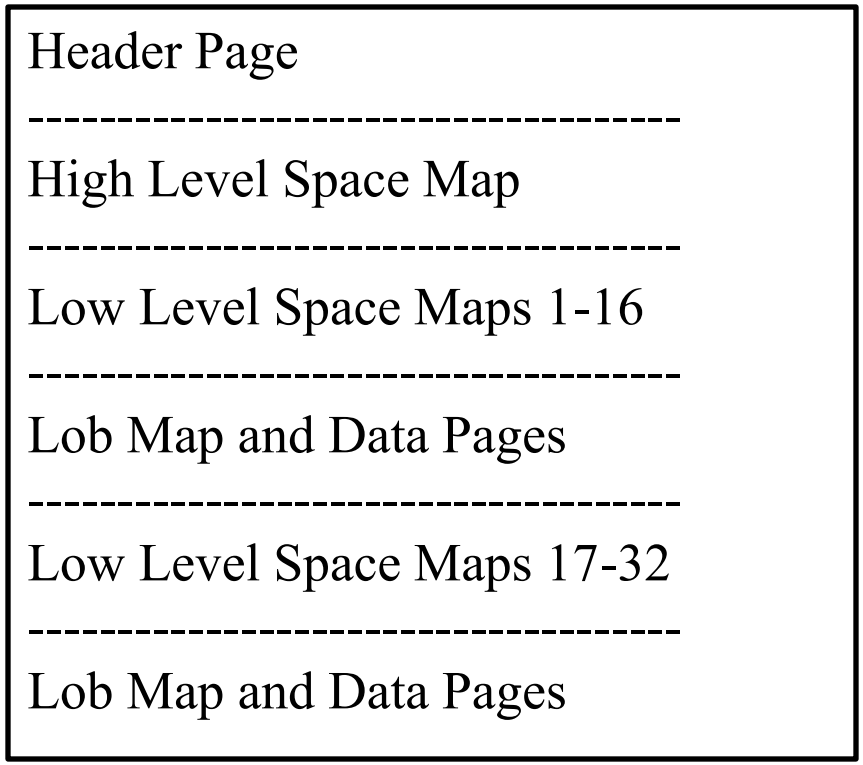
Standard UPDATE behavior



LOB SPACE – Format

- Header
- High Level Space Map
- Low Level Space Map
- LOB Map and Data Pages

UPDATES induce a delete of the before image and insert of the after image



DB2 Catalog Tables affected by INLINE LOBs

- SYSIBM.SYSDATATYPES -- new column INLINE_LENGTH
  value of -1 = no INLINE LENGTH
  value greater than or equal zero = inline length attribute (in byte) of the type if it is based on a LOB source type

- SYSIBM.SYSCOLUMNS – column LENGTH. The number does not include the internal prefixes that are used to record the actual length and null state, where applicable.
  value > 4 = value of INLINE LENGTH + 4. The maximum length of the LOB column is found in column LENGTH2

- SYSIBM.SYSCOPY -- column ICTYPE=A and column STYPE=I
  for Alter INLINE LENGTH activity
  TTYPE=I REORG INCREASE INLINE LENGTH
  TTYPE=D REORG DECREASE INLINE LENGTH

DB2 Catalog Tables affected by INLINE LOBs

- SYSIBM.SYSTABLEPART – CHECKFLAG – new values

  D - inline length of LOB column was decremented with an alter

  I – inline length of LOB column was incremented with an alter

DB2 Directory Tables affected by INLINE LOBs

- DSNDB01.SPT01 – now a Partition-by-Growth space

- ZPARM - SPT01_INLINE_LENGTH
  Max inline length of LOB column in SPT01
  valid values = NOINLINE, 1 – 32138
  default value is 32138
  Use the largest possible setting to decrease fetches from auxiliary tables and increase compression efficiency

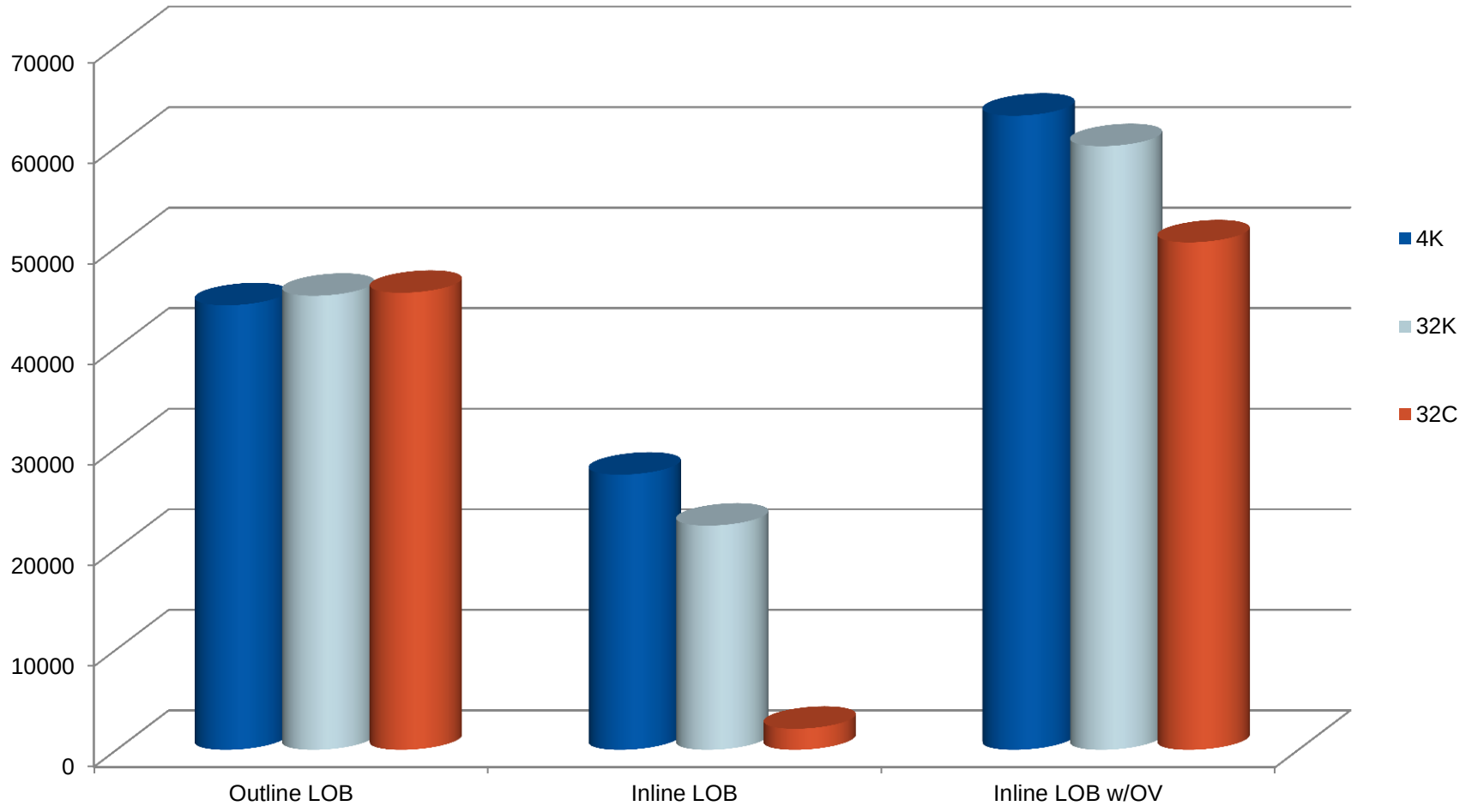
- ZPARM - COMPRESS_SPT01
  specifies whether the SPT01 table space is to be compressed.
  valid values = NO, YES
  default is NO

All data sharing members must use the same settings for these parms

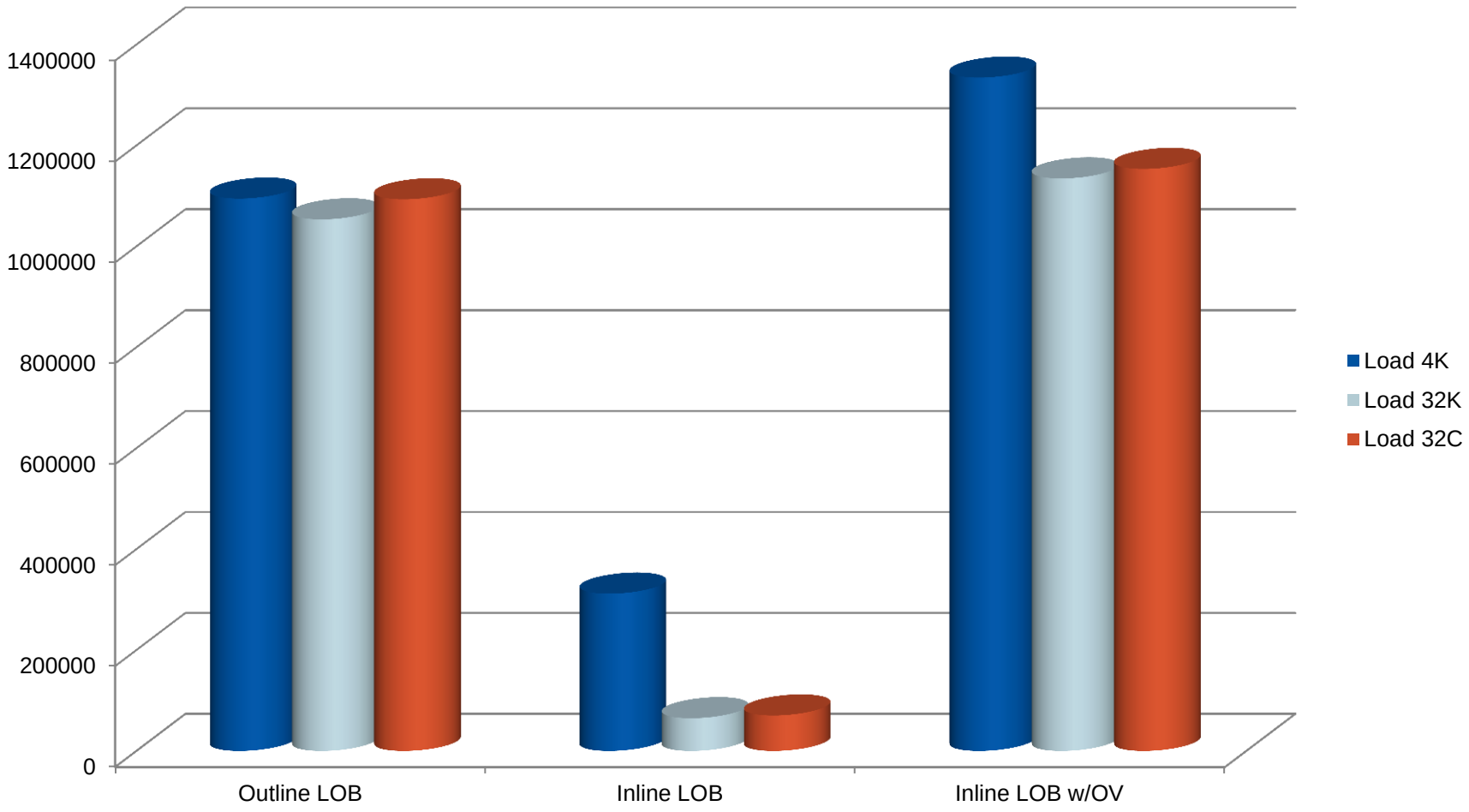
My experiment – objects and data

- Three object types
  CLOB(2000) (outline lob)
  CLOB(2000) INLINE LENGTH 1600 (no overflow, index on expression)
  CLOB(2000) INLINE LENGTH 1000 (overflow, index on expression)
- Used different page sizes -- 4K,32K, 32K compressed
- Columns in table: LOB, integer, timestamp
- 500,000 rows
- LOB data is 1500 bytes
- Measured EXCP in MSTR, DBM1, DIST, IRLM

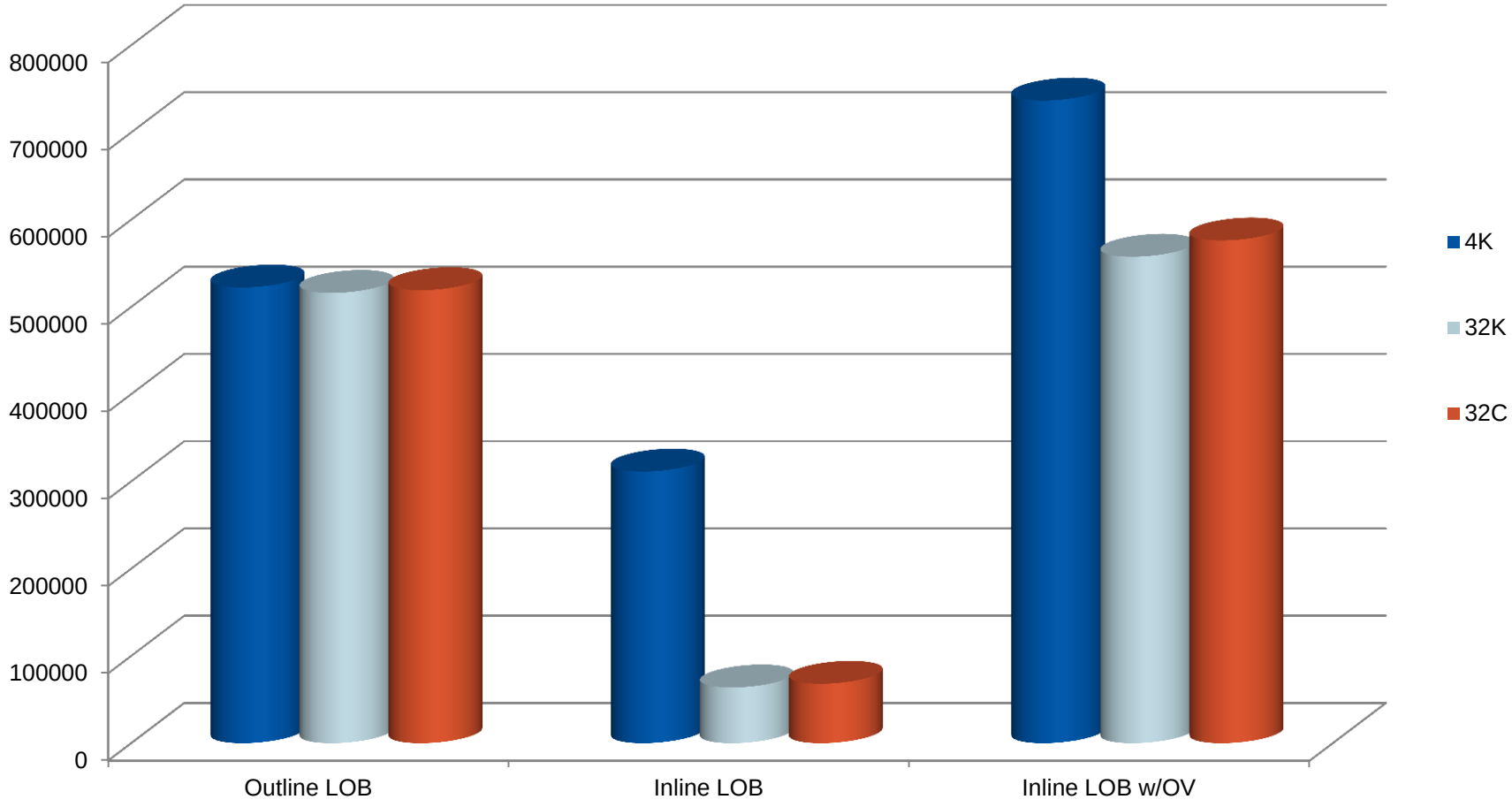
Dasd (in tracks)



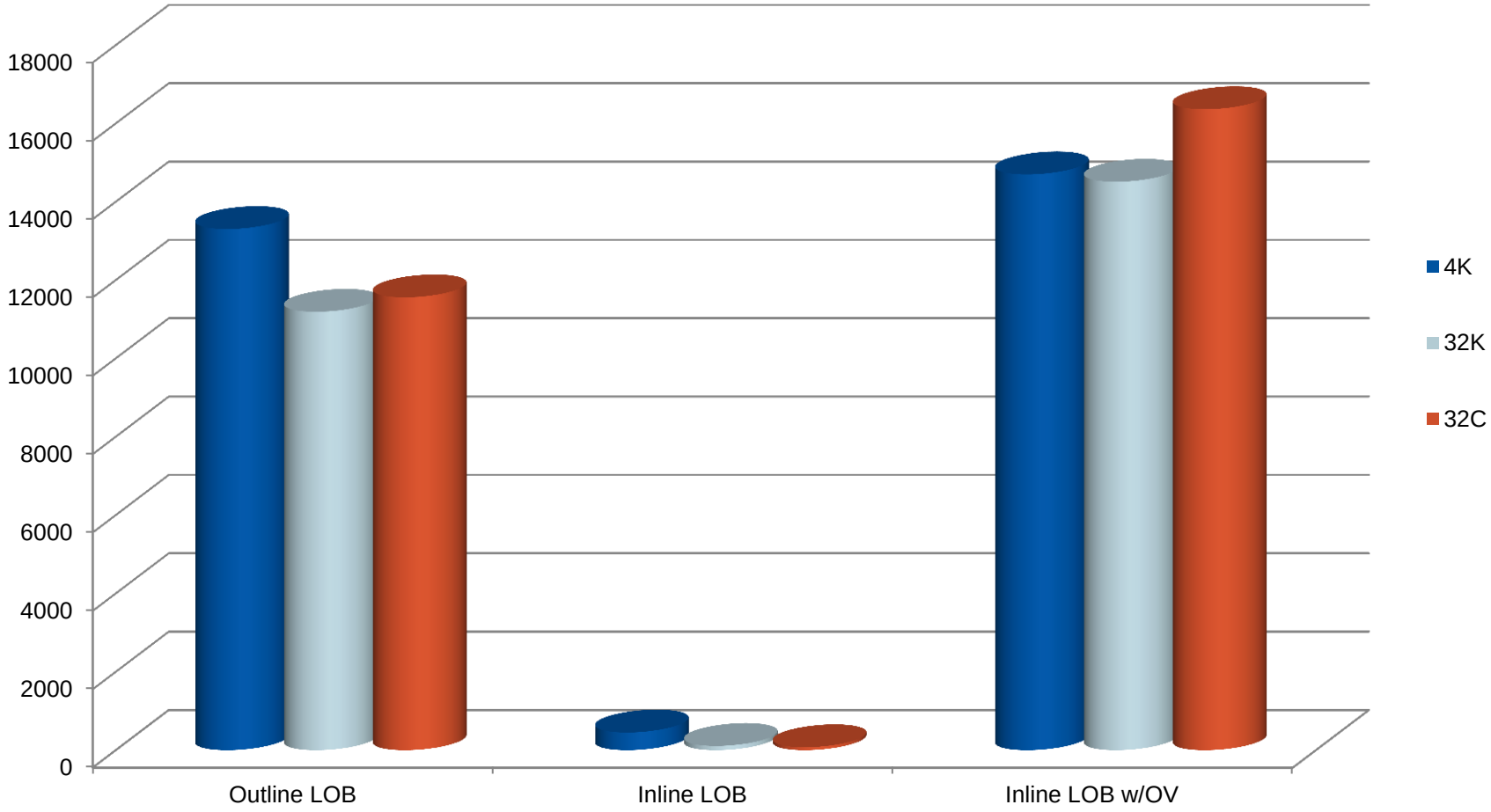
LOAD EXCP


1400000
1200000
1000000
800000
600000
400000
200000
0
Outline LOB
Inline LOB
Inline LOB w/OV
Load 4K
Load 32K
Load 32C

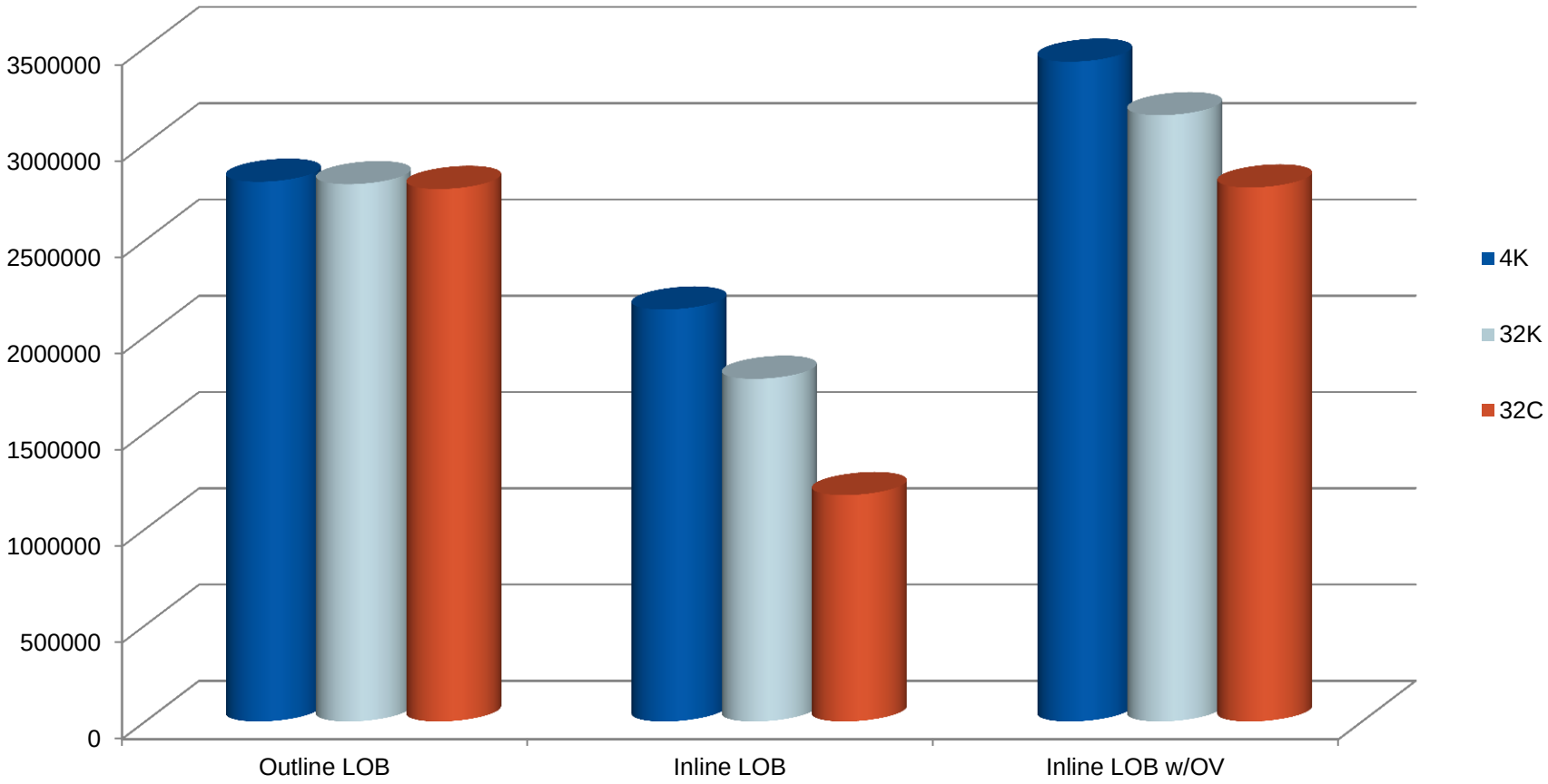
COPY EXCP


800000
700000
600000
500000
400000
300000
200000
100000
0
Outline LOB
Inline LOB
Inline LOB w/OV
4K
32K
32C

DELETE EXCP


18000
16000
14000
12000
10000
8000
6000
4000
2000
0
Outline LOB
Inline LOB
Inline LOB w/OV
4K
32K
32C

INSERT EXCP


3500000
3000000
2500000
2000000
1500000
1000000
500000
0
Outline LOB
Inline LOB
Inline LOB w/OV
4K
32K
32C

Summary

- There is no definitive answer as to whether using INLINE LOBs will benefit your shop. It depends on your data and usage trends.

Bibliography

- DB2 10 for z/OS SQL Reference, SC19-2983-02
- DB2 10 for z/OS Utility Guide and Reference, SC19-2984-02
- DB2 V10 Performance Topics

http://www.redbooks.ibm.com/Redbooks.nsf/RedbookAbstracts/sg247942.html?Open

Sandi Smith
BMC Software Inc.
sandi_smith@bmc.com

E10
DB2 10 Inline LOBs