Servicio Andaluz de Salud
CONSEJERÍA DE SALUD

BEST PRACTICES EL USO DE COMPRESIÓN DE ÍNDICES EN BASE DE DATOS ORACLE.

Referencia documento:
InfV5_JASAS_CompressionIndices_BestPractices_V920.doc
Fecha: 13 de noviembre de 2018
Versión: 9.2.0

Registro de Cambios

Fecha	Autor	Versión	Notas
16 de Junio de 2016	Isidro Granados	7.1	Version Inicial
14 de octubre de 2016	Isidro Granados	7.2.0	Revisión de Noviembre de 2016, contrato 2014-2016
14 de junio de 2017	Isidro Granados	8.1.0	Revisión de Junio de 2017, contrato 2016-2018
16 de noviembre de 2017	Isidro Granados	8.2.0	Revisión de Noviembre de 2017, contrato 2016-2018
16 de junio de 2018	Isidro Granados	9.1.0	Revisión de junio de 2018, contrato 2016-2018
13 de noviembre de 2018	Isidro Granados	9.2.0	Revisión de noviembre de 2018, contrato 2016-2018

Revisiones

Nombre	Role
Oracle ACS Engineers	ACS Service Engineer
Jose Maria Gomez	ACS Service TAM

Distribución

Copia	Nombre	Empresa
1	Subdirección de Tecnologías de la Información	Consejería de Salud, Junta de Andalucía
2	Servicio de Coordinación de Informática de la Consejería de Hacienda y Administración Pública	Consejería de Hacienda y Administración Pública, Junta de Andalucía

Índice de Contenidos

Control de cambios

Cambio	Descripción	Página
1	No se realizan cambios en esta versión del documento.	N/A

Introducción

Oracle RDBMS dispone de dos opciones e compresión para índices llamadas "Prefix compression" (antes llamada "Key compression") y "Advanced Index Compress" que se habilita en la creación o reconstrucción del índice con la cláusula `COMPRESS`.

"Prefix Compression" viene incluida dentro de Oracle Database Enterprise Edition sin coste extra, en cambio "Advanced Index Compress" que está disponible a partir de versión 12.1.0.2 sí requiere licenciamiento extra de la opción de Advanced Compression Option.

El presente informe muestra las características más significativas de la compresión de índices en la base de datos Oracle del tipo "Prefix Compression" mostrando cómo determinar cuándo la compresión puede ser beneficiosa en un índice.

El informe está basado en versión 12cR2 (12.2.0.1), aunque muchos de los conceptos indicados son válidos para versiones anteriores.

El informe mostrará en un punto independiente una introducción a la opción de coste extra **"**Advanced Index Compression".

Compresión de índices del tipo "Prefix compression"

La compresión de índices "Prefix compression" (o "Key compression") es aplicable a índices B-tree y a Tablas Organizadas por índices (IOTs). No aplica a índices Bitmap, estos son guardados de forma comprimida por defecto.

La idea detrás del "prefix compression" es eliminar valores repetidos. En muchos índices, los valores clave son a menudo repetidos. Valores clave repetidos en índices no únicos suelen ser bastantes corrientes, o en índices únicos cuando hay varias columnas en el índice. En estos casos se puede salvar espacio evitando tener que guardar estos valores duplicados en el índice.

¿En qué consiste la compresión de índices?

La compresión de índices "Prefix compression" se habilita en la creación o reconstrucción del índice añadiendo la cláusula `COMPRESS <RANGO>` al comando de `CREATE o ALTER INDEX REBUILD`. Para modificar un índice ya creado para habilitar o deshabilitar la compresión habrá que reconstruir el índice.

La compresión es realizada a nivel de bloque (a nivel de leaf blocks), esto es, bloque a bloque de forma independiente. La compresión se realiza dividiendo las claves en 2 partes: la "prefix part" (parte prefija) y "sufix part" (parte sufija). Cuando se usa la cláusula `COMPRESSS` se le indicará la longitud del prefijo (rango o level), esto será el número de columnas prefijas a comprimir. Si no se indica al `COMPRESS` ningún rango, Oracle tomará como longitud del prefijo:

- Para índices únicos, todas las columnas menos una.
- Para índices no-únicos, todas la columnas.

La parte prefija es compartida por todos los sufijos del bloque del índice.

Por ejemplo, imaginemos un índice compuesto sobre la tabla oe.orders:

```
CREATE INDEX orders_mod_stat_ix ON orders ( order_mode,
order_status );
```

Al no estar comprimido, un leaf block del índice irá guardando entradas con sus claves y el rowid de la fila en la tabla correspondiente, por ejemplo:

```
online,0,AAAPvCAAFAAAAFaAAa
online,0,AAAPvCAAFAAAAFaAAg
online,0,AAAPvCAAFAAAAFaAAl
online,2,AAAPvCAAFAAAAFaAAm
online,3,AAAPvCAAFAAAAFaAAq
online,3,AAAPvCAAFAAAAFaAAt
```

Si el índice fuera creado con la opción de COMPRESS 1:

```
CREATE INDEX orders_mod_stat_ix ON orders ( order_mode,
order_status )COMPRESS 1;
```

La parte prefija correspondería a la columna order_mode y el sufijo a order_status,rowid. Conceptualmente, la base de datos realiza la compresión guardando las entradas en el bloque del índice de esta forma:

```
online
0,AAAPvCAAFAAAAFaAAa
0,AAAPvCAAFAAAAFaAAg
0,AAAPvCAAFAAAAFaAAl
2,AAAPvCAAFAAAAFaAAm
3,AAAPvCAAFAAAAFaAAq
3,AAAPvCAAFAAAAFaAAt
```

De esta forma se van evitando ocurrencias duplicadas de "`online`", que será compartida por todas las entradas del índice en el bloque.

Si se hubiera creado el índice con COMPRESS sin indicar rango, esto es:

```
CREATE INDEX orders_mod_stat_ix ON orders ( order_mode,
order_status )COMPRESS;
```

Cogerá en este caso el valor por defecto a 2, y la parte prefija consistirá en la concatenación de los valores de `order_mode` y `order_status`. Esto es, los valores de claves duplicadas de prefijos como `online,0` y `online,3` serían comprimidos. Conceptualmente, la base de datos realiza la compresión guardando las entradas en el bloque del índice de esta forma:

```
online,0
AAAPvCAAFAAAAFaAAa
AAAPvCAAFAAAAFaAAg
AAAPvCAAFAAAAFaAAl
online,2
AAAPvCAAFAAAAFaAAm
online,3
AAAPvCAAFAAAAFaAAq
AAAPvCAAFAAAAFaAAt
```

Cuantos más valores distintos haya de las columnas comprimidas, más entradas tendrá la parte prefija (prefix table). Cuando menos valores distintos tengan las columnas comprimidas menos entradas tendrá la parte prefija (prefix table). Generalmente, cuantas menos entradas tenga la parte prefija mejor la compresión.

Ventajas e inconvenientes de utilizar "Prefix compression"

Ventajas:

- Reducir espacio ocupado por el índice. Depende de los datos guardados en el índice y del rango de compresión el espacio liberado puede ser más o menos significativo.
- Al tener menos leaf blocks en el índice:

- Potencialmente se reducirán el número de lecturas lógicas y físicas cuando se accede a través del índice en un index range / index (fast) full scans y por tanto se mejorará el rendimiento.
- Esto también mejorará la eficiencia del buffer cache.
- Desde el punto de vista del Optimizador basado en coste (CBO) se potencialmente podrá generar un plan de ejecución con menos coste.

Inconvenientes:

- Se tiene que hacer un análisis para determinar cuál es el rango o nivel óptimo de compresión, ya que se podría hasta obtener un índice más grande comprimido que el mismo índice sin usar compresión (compresión negativa). Al comprimir se añade un overhead en el almacenamiento del índice ya que cada entrada en la parte prefija tiene un overhead de 4 bytes. Si este overhead no se compensa con la liberación de valores duplicados no será un buen nivel o un buen índice candidato a comprimir.
- Puede incrementar la contención ya que habrá más filas por leaf block en el índice.
- Puede requerir más CPU para acceder a la clave, ya que Oracle tendrá que hacer la traslación de la parte <prefix, suffix> a la clave correspondiente cuando se haga un index scan.
- Al comprimir el plan de ejecución de las sentencias podrían cambiar.

En resumen, dependiendo del tipo de datos en el índice y las operativas en la base de datos la compresión de índices puede ayudar a mejorar el rendimiento en las aplicaciones y a la vez liberar espacio ocupado. También se pueden encontrar situaciones como las indicadas arriba en las que la compresión penalice el rendimiento, por tanto, es recomendado probar en un entorno de pruebas con datos similares el rendimiento de las aplicaciones tras la compresión de índices, así como medir la liberación real de espacio en la compresión de índices.

¿Cuándo comprimir?

Se pueden comprimir índices que no sean únicos o únicos que tengan al menos 2 columnas.

La compresión de índices será muy beneficiosa cuando las columnas iniciales del índice tienen muchos valores repetidos dentro de un leaf block, y cuanto más grande el índice más espacio ocupado se liberará. Sin embargo, cuando las primeras columnas del índice tienen pocos o ningunos valores repetidos en un índice, tendremos un problema ya que se creará una parte prefija muy grande, que tendrá de todas formas la mayoría de los valores de las columnas. En tal caso las entradas del índice apuntarán a entradas de la parte prefija que apenas son compartidas (si acaso) con otras entradas del índice.

Por otro lado, en caso de que el índice no tenga unos datos uniformes en las columnas comprimidas, esto es, haya leaf blocks con datos con valores muy repetidos buenos candidatos a ser comprimidos y leaf blocks con valores muy distintos, key compression actuará sobre todos los bloques con el mismo rango, comprimiendo algunos significativamente y en otros bloques no. Para solucionar esta casuística en versión 12c se puede usar la nueva opción Advanced Index Compression, Oracle automáticamente determinará qué bloques serán beneficiosos y cuáles no.

Buscando el valor óptimo del COMPRESS level para compresión de índices

Como se ha comentado anteriormente es importante el analizar el rango o nivel óptimo a indicar en la cláusula de `COMPRESS`.

A partir de versión 9i, se pueden usar las columnas `opt_cmpr_count`, `opt_cmpr_pctsave` de `INDEX_STATS` para conocer a priori cual es este valor antes de comprimir el índice.

Para completar la vista `INDEX_STATS` previo hay que lanzar un:

```
ANALYZE INDEX <schema.index_name> VALIDATE STRUCTURE;
```

Nota:

Esto es igual que

```
ANALYZE INDEX <schema.index_name> VALIDATE STRUCTURE OFFLINE;
```

Mientras se está ejecutando no se permiten de forma concurrente sentencias `INSERT, UPDATE, y DELETE` sobre los objetos referenciados, únicamente se permiten consultas. Si se usa opción `ONLINE` se podrán realizar operaciones concurrentes pero no se completará la vista `INDEX_STATS`.

Tras el `analyze validate structure` se podrán consultar las siguientes columnas `opt_cmpr_count, opt_cmpr_pctsave`:

```
SELECT name, partition_name, blocks, opt_cmpr_count,
opt_cmpr_pctsave FROM INDEX_STATS;
```

- opt_cmpr_count: Indicará cuantas columnas pueden ser comprimidas en el índice para obtener el máximo beneficio del "key compression"
- opt_cmpr_pctsave: Indicará el porcentaje de espacio ahorrado si se indica el valor anterior como parámetro en la reconstrucción del índice.

Esto es, el valor `opt_cmpr_count` será lo que usaremos para comprimir el índice:

```
ALTER INDEX <schema.index_name> REBUILD ONLINE COMPRESS
<opt_cmpr_count>;
```

Compresión con índices particionados

Se pueden comprimir varias o todas las particiones de un índice B-tree usando la key compression.

En el siguiente ejemplo se muestra un índice local particionado donde todas sus particiones excepto la más reciente están comprimidas:

```
CREATE INDEX i_cost1 ON costs_demo (prod_id) COMPRESS LOCAL
   (PARTITION costs_old, PARTITION costs_q1_2003,
    PARTITION     costs_q2_2003,     PARTITION     costs_recent
    NOCOMPRESS);
```

Si se quiere usar compresión en un índice particionado, se tiene que habilitar la compresión a nivel de índice. Luego posteriormente habilitar o deshabilitar la compresión individualmente para cada partición del índice particionado. Se puede habilitar y deshabilitar la compresión haciendo rebuild de cada partición.

No se puede especificar COMPRESS (o NOCOMPRESS) explicitamente para una subpartición de un índice. Todas las subparticiones de un índice para una partición concreta heredarán la configuración de compresión indicada a la partición a la que pertenecen.

Para modificar el atributo key compression para todas las subparticiones de una partición, primero habrá que lanzar un ALTER INDEX...MODIFY PARTITION y posteriormente hacer un rebuild de todas las subparticiones.

> Nota: La clausula MODIFY PARTITION marcará todas las subparticiones del índice como UNUSABLE.

La nota de MyOracleSupport *Note: 179950.1 Handling of compress option on partitioned indexes,* muestra diferentes uso de la opción COMPRESS con índices particionados.

Ejemplo de compresión de índice

A continuación se muestra con un ejemplo cómo valorar cuando la compresión de un índice reduciría espacio comparando los distintos rangos en la cláusula `COMPRESS`.

1. Creación de la tabla y carga de datos ejemplo:

```
conn oss/oss

drop table objetos;

create table objetos as select
OWNER,OBJECT_NAME,OBJECT_ID,OBJECT_TYPE,CREATED,STATUS from
dba_objects;
```

```
INSERT INTO OBJETOS select
OWNER,'A'||OBJECT_NAME,OBJECT_ID,OBJECT_TYPE,CREATED,STATUS from
objetos;
COMMIT;

INSERT INTO OBJETOS select
OWNER,'B'||OBJECT_NAME,OBJECT_ID,OBJECT_TYPE,CREATED,STATUS from
objetos;
COMMIT;

INSERT INTO OBJETOS select
OWNER,'C'||OBJECT_NAME,OBJECT_ID,OBJECT_TYPE,CREATED,STATUS from
objetos;
COMMIT;

INSERT INTO OBJETOS select
OWNER,'D'||OBJECT_NAME,OBJECT_ID,OBJECT_TYPE,CREATED,STATUS from
objetos;
COMMIT;

INSERT INTO OBJETOS select
OWNER,'E'||OBJECT_NAME,OBJECT_ID,OBJECT_TYPE,CREATED,STATUS from
objetos;
COMMIT;
```

2. Creación de un índice no-único compuesto sin compresión y se analizan datos:

```
create index objetos_idx on objetos (owner,object_type,object_name);


select num_rows,blevel,leaf_blocks, compression, PREFIX_LENGTH from
dba_indexes where index_name='OBJETOS_IDX' and owner='OSS';


  NUM_ROWS     BLEVEL LEAF_BLOCKS COMPRESS PREFIX_LENGTH
---------- ---------- ----------- -------- -------------
   2780736          2       20454 DISABLED


ANALYZE INDEX oss.OBJETOS_IDX VALIDATE STRUCTURE;

SELECT name, blocks, lf_blks, opt_cmpr_count, opt_cmpr_pctsave FROM
INDEX_STATS;

NAME         BLOCKS    LF_BLKS    OPT_CMPR_COUNT  OPT_CMPR_PCTSAVE
------------ --------- ---------- --------------- ---------------
OBJETOS_IDX  21504     20454      2               27

SQL>  select  sum(bytes)/1024/1024  "MB"  from  dba_segments  WHERE
SEGMENT_NAME='OBJETOS_IDX' AND OWNER='OSS';

        MB
----------
       168
```

En este caso, el índice sin comprimir ocupa 168MB con 20454 leaf blocks. Tras lanzarle el `analyze validate structure` nos indica que tendríamos una reducción del 27% si se comprimiera con rango 2.

En este caso, la tercera columna clave del índice OBJECT_NAME tiene muchos distintos valores, el que esté esta columna en la parte prefija no sería recomendable para la compresión como veremos a continuación.

3. Compresión del índice sin indicar rango y se analizan datos:

```
ALTER INDEX oss.OBJETOS_IDX REBUILD COMPRESS;

select  num_rows,blevel,leaf_blocks,compression,  PREFIX_LENGTH  from
dba_indexes where index_name='OBJETOS_IDX' and owner='OSS';

  NUM_ROWS     BLEVEL LEAF_BLOCKS COMPRESS PREFIX_LENGTH
---------- ---------- ----------- -------- -------------
   2780736          2       22740 ENABLED              3

SQL>  select  sum(bytes)/1024/1024  "MB"  from  dba_segments  WHERE
SEGMENT_NAME='OBJETOS_IDX' AND OWNER='OSS';

        MB
----------
       184

SQL> ANALYZE INDEX oss.OBJETOS_IDX VALIDATE STRUCTURE;


SQL> SELECT name, blocks, lf_blks, opt_cmpr_count, opt_cmpr_pctsave
FROM INDEX_STATS;

NAME         BLOCKS    LF_BLKS    OPT_CMPR_COUNT  OPT_CMPR_PCTSAVE
------------ --------- ---------- --------------- ---------------
OBJETOS_IDX  23552     22740      2               33
```

Vemos que el `COMPRESS` por defecto del índice provoca una compresión negativa, esto es, hace que crezca el índice a 184MB. Internamente utiliza una longitud del prefijo de 3, esto es, es similar a haber lanzado un COMPRESS 3. El `analyze validate structure` sigue indicando cómo el más optimo el rango 2.

4. Compresión del índice poniendo una longitud del prefijo de 1 y se analizan datos:

```
ALTER INDEX oss.OBJETOS_IDX REBUILD COMPRESS 1;

select  num_rows,blevel,leaf_blocks,compression,  PREFIX_LENGTH  from
dba_indexes where index_name='OBJETOS_IDX' and owner='OSS';
```

```
  NUM_ROWS     BLEVEL LEAF_BLOCKS COMPRESS PREFIX_LENGTH
---------- ---------- ----------- -------- -------------
   2780736          2       18248 ENABLED              1


SQL>  select  sum(bytes)/1024/1024  "MB"  from  dba_segments  WHERE
SEGMENT_NAME='OBJETOS_IDX' AND OWNER='OSS';

        MB
----------
       152

SQL> ANALYZE INDEX oss.OBJETOS_IDX VALIDATE STRUCTURE;

SQL> SELECT name, blocks, lf_blks, opt_cmpr_count, opt_cmpr_pctsave
FROM INDEX_STATS;

NAME         BLOCKS    LF_BLKS    OPT_CMPR_COUNT  OPT_CMPR_PCTSAVE
------------ --------- ---------- --------------- ----------------
OBJETOS_IDX  19456      18248     2               18
```

Vemos que el `COMPRESS 1` reduce a 152MB de los 168MB iniciales el tamaño del índice.

5. Compresión del índice poniendo una longitud del prefijo de 2 y se analizan datos:

```
ALTER INDEX oss.OBJETOS_IDX REBUILD COMPRESS 2;

select  num_rows,blevel,leaf_blocks,compression,  PREFIX_LENGTH  from
dba_indexes where index_name='OBJETOS_IDX' and owner='OSS';

  NUM_ROWS     BLEVEL LEAF_BLOCKS COMPRESS PREFIX_LENGTH
---------- ---------- ----------- -------- -------------
   2780736          2       14873 ENABLED              2

SQL>  select  sum(bytes)/1024/1024  "MB"  from  dba_segments  WHERE
SEGMENT_NAME='OBJETOS_IDX' AND OWNER='OSS';

        MB
----------
       120

SQL> ANALYZE INDEX oss.OBJETOS_IDX VALIDATE STRUCTURE;

SQL> SELECT name, blocks, lf_blks, opt_cmpr_count, opt_cmpr_pctsave
FROM INDEX_STATS;

NAME         BLOCKS    LF_BLKS    OPT_CMPR_COUNT  OPT_CMPR_PCTSAVE
------------ --------- ---------- --------------- ----------------
OBJETOS_IDX  15360     14873      2                0
```

Vemos que el `COMPRESS 2` reduce a 120MB de los 168MB iniciales el tamaño del índice:

Resumen comparativo de la compresión del índice `OBJETOS_IDX`:

Tamaño (MB)	Leaf Blocks	Opción compresión	Reducción espacio (%)	Notas
168	20454	No COMPRESS		
184	22740	COMPRESS	-11,17%	Por defecto usa COMPRESS 3
152	18248	COMPRESS 1	10,78%	
120	14873	COMPRESS 2	27,2%	Coincide con el valor óptimo estimado por ANALYZE (27%)

Compresión de índices del tipo "Advanced Index Compression"

En versión 12.1.0.2 se introdujo una nueva funcionalidad llamada "advanced index compression" que se habilitada con la opción `COMPRESS ADVANCED` en el `CREATE o ALTER INDEX REBUILD`.

Esta nueva funcionalidad facilita y mejora la forma en la que los índices son comprimidos respecto a la opción de compresión "prefix compression" pero requiere licenciamiento extra de la opción de Advanced Compression Option.

Advanced index compression mejora los ratios de compresión de forma significativa permitiendo un eficiente acceso a los índices. Advanced index compression funciona correctamente en todos los tipos soportados de índices, incluyendo índices que no son buenos candidatos para la key compression. Advanced index compression no está soportado para índices tipo bitmap ni para tablas organizadas por índice.

Al igual que prefix compression, advanced index compression se puede especificar para índices no-únicos y únicos con más de una columna.

En advanced index compression en lugar de usar para todos los bloques un valor fijo de longitud de claves a eliminar, usa un mecanismo adaptable a cada bloque de forma que la base de datos automáticamente elige la mejor compresión para cada bloque, así el usuario no tiene por qué requerir conocer las características de los datos para que la compresión sea efectiva.

En versión 12.2.0.1 existen dos opciones:

`COMPRESS ADVANCED LOW:`

Este nivel comprime menos el índice que HIGH, pero permite un acceso más rápido al índice con un mínimo sobrecoste de CPU. Requiere un compatible igual o mayor a 12.1.0.

`COMPRESS ADVANCED HIGH:`

Este nivel comprime más el índice que LOW al utilizar un algoritmo de compresión más complejo, pero provocará un acceso más lento al índice al añadir algún sobrecoste de CPU. Sin embargo al comprimir más el índice podrá mejorar el rendimiento de sentencias que accedan al índice al necesitar leer menos bloques. Requiere un compatible igual o mayor a 12.2.0. Es la opción por defecto para el COMPREESS ADVANCED si se omite LOW o HIGH

Ejemplo de creación:

```
CREATE INDEX hr.emp_mndp_ix
  ON hr.employees(manager_id, department_id)
  COMPRESS ADVANCED;
```

Para comprobar el tipo de compresión:

```
SELECT COMPRESSION FROM DBA_INDEXES WHERE INDEX_NAME
='EMP_MNDP_IX';
```

```
COMPRESSION
-------------
ADVANCED HIGH
```

En versión 12c para estimar el ratio de compresión de la "Advanced index compression se puede usar el procedure `GET_COMPRESSION_RATIO` del package `DBMS_COMPRESSION`, con las opciones `COMP_INDEX_ADVANCED_HIGH` y `COMP_INDEX_ADVANCED_LOW`. Ver ejemplos de uso en las notas de MOS: *NOTE:1911547.1 - How to estimate COMPRESSION RATIO for Indexes in 12.1* y *NOTE: 1957911.1 How to Estimate Compression Ratio of all the Indexes of a Table in 12c?*

Comparando la reducción de espacio en el ejemplo mostrado en la compresión tipo "prefix compression" con el "Advanced Index Compression":

Tamaño (MB)	Opción compresión	Reducción espacio (%)
168	No COMPRESS	
184	COMPRESS	-11,17%
152	COMPRESS 1	10,78%
120	COMPRESS 2	27,2%
120	COMPRESS ADVANCED LOW	27,2%
52	COMPRESS ADVANCED HIGH	70%