

Компрессия и голденгейт

Представим ситуацию. У нас есть табличка, которая содержит информацию о заказах. Когда-то заказов в табличке были сотни, потом тысячи, десятки тысяч, и в какой-то момент оказалось, что заказов у нас миллиарды, а бизнес не хочет удалять исторические данные. При этом место у нас не резиновое и стоит дорого.

Тогда мы вспоминаем, что Oracle предоставляет неплохой инструмент для сжатия данных в таблицах — от классических Advanced/OLTP/Basic до специфических компрессий, которые есть на Exadata: FOR QUERY HIGH/LOW и FOR ARCHIVE HIGH/LOW. И это хорошо.

Но в таблицу идёт активная вставка данных (ведь у нас очень много покупателей и заказов), плюс на таблице выполняются операции обновлений, потому что заказы могут меняться, отменяться и т.д. Поэтому, если мы просто сожмём таблицу на нашей замечательной Exadata, мы, конечно, получим выигрыш в месте (возможно, даже 20-кратный), но при этом полностью «убьём» приложения, которые изменяют таблицу.

Остановимся на том, что наша таблица заполняется на некоем хранилище репликатом GoldenGate. С таблицей работает координированный репликат, потому что если не распараллеливать вставки и обновления, то у нас начинает накапливаться отставание.

Как только мы сожмём таблицу, наш координированный репликат начнёт выдавать ошибку ORA-00060 (deadlock detected) и падать на взаимоблокировках: одновременно несколько потоков пытаются изменить один и тот же блок.

Но выход есть. Надо таблицу партиционировать. Мы предполагаем, что заказы старше определённой даты уже не могут быть отменены или изменены. Мы партиционируем таблицу по интервалу в один месяц, сжимаем «старые» партиции, которые не будут меняться, той компрессией, которую мы выбрали, а новые партиции оставляем как есть — несжатыми.

По мере заполнения партиций мы можем автоматически сжимать устаревающие партиции. Для этого нам надо предусмотреть, чтобы индексы (для работы репликата нужен уникальный индекс в любом случае) были локальными.

Пример реализации

Создадим таблицу заказов, партиционированную по диапазону дат с интервалом в один месяц:

```

CREATE TABLE orders (
  order_id    NUMBER PRIMARY KEY,
  order_date  DATE,
  customer_id NUMBER,
  order_status VARCHAR2(20),
  ...
)
PARTITION BY RANGE (order_date) INTERVAL (INTERVAL '1' MONTH)
(
  PARTITION p_historical VALUES LESS THAN (TO_DATE('2020-01-01','YYYY-MM-DD'))
);

```

Затем создадим локальный уникальный индекс на поле `order_id`, необходимый для работы репликации:

```

CREATE UNIQUE INDEX orders_pk ON orders (order_id, order_date) LOCAL;

```

Теперь можно сжать партицию, данные в которой уже не меняются. Например, для партиции за январь 2020 года:

```

ALTER TABLE orders
  MOVE PARTITION p_2020_jan
  PARALLEL 16
  COMPRESS FOR ARCHIVE HIGH;

SELECT 'ALTER INDEX ' || INDEX_NAME || ' REBUILD PARTITION ' || PARTITION_NAME || ' PARALLEL 16;'
FROM DBA_IND_PARTITIONS IP
WHERE IP.STATUS = 'UNUSABLE';

```

Это сжатие не затронет активные партиции, и репликат продолжит работать без конфликтов за блоки.

Таким образом, партиционирование в сочетании с выборочным сжатием позволяет эффективно экономить место, не жертвуя производительностью операций DML и не создавая проблем для параллельной репликации.

Revision #2

Created 24 February 2026 14:58:43 by catmaster

Updated 25 February 2026 10:26:03 by catmaster