表の断片化が進んでいる、索引が肥大化している判断の基準

表の断片化、索引の肥大化を調査するＳＱＬ

【 注　意 】

以下の SQL を使用して、表と索引の断片化を調査する時は、事前にオブジェクトの統計情報を最新の状態にしてください

『表の複数ブロックへの拡張』と『表の断片化』の違い

表のデータが順次増加していき、複数ブロックに拡張されていく分には、１レコードの内容が、１ブロックに収まるか、もしくは、連続する複数ブロックにデータが格納される

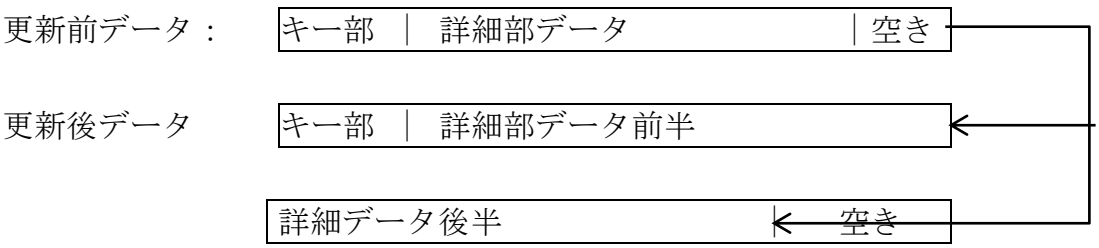
これが、通常拡張である

この場合、PCTFREE で設定した設定値により、空き領域の余裕を持たせてデータを保存していく

追加データ：　キー部 ｜ 詳細部データ ｜空き

〃　後半データ：　詳細部データ前半 ｜空き

これに対して、断片化とは、データ更新が行われた結果、元々あったブロックだけでは収まり切れなくなって、不連続なデータ・ブロックに分割されてデータが格納されることを云う



このように、１つのレコードが、不連続なブロックに格納されている状態では、１レコードの読み込みに対して、複数ブロックの読み込みが必要となる

このような状態が頻繁になると、データ処理に対するブロック読み込み数が大きくなり、処理時間が長時間になってしまう

このようなことが無いように、断片化したテーブルは再編成を行い、断片化が無い状態に保つ必要がある

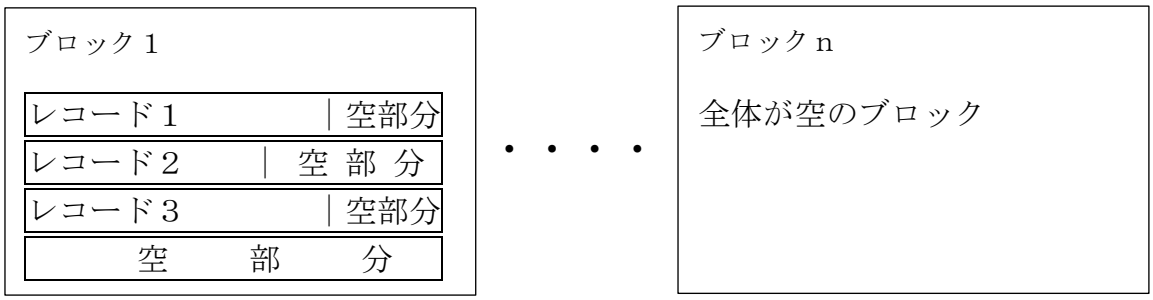
断片化が進んだテーブル（表）へのメンテナンス

テーブルの断片化率が７０％以下となっているテーブルは、断片化解消のメンテナンスのために、テーブル再編成が必要である

【注意点】

前ページに記述したような、１つのレコードが不連続なディスク・ブロックに跨ってデータ保存されているようなことを示す指標は、Oracle のオブジェクト統計情報にはない

なので、Oracle の表に対する断片化を判断する場合には、物理的に確保されたブロックの中の空領域がどれくらいあるかを見て、断片化率としている



表断片化率の計算方法

SUM 関数を使用しているのは、オブジェクトが複数のパーティションに分けられてエリア定義されていた場合を想定して合計している

【参考情報】

A について

```
SELECT  owner,  segment_name,  SUM ( bytes ) seg_bytes
    FROM  dba_segments
    WHERE  owner  =  'KOZUE'
    GROUP BY  owner ,  segment_name ;
```

実際のデータの合計長

OWNER	SEGMENT_NAME	SEG_BYTES
KOZUE	EMP	107946

B について

```
SELECT  owner ,  table_name ,  SUM ( num_rows * avg_row_len )
          tab_bytes
    FROM  dba_tables
    WHERE  owner  = 'KOZUE'
    GROUP BY  owner,  table_name ;
```

論理上のデータ長の合計

OWNER	TABLE_NAME	TAB_BYTES
KOZUE	EMP	87935 ＝2045 バイト*43 件

断片化率（簡易的な）について

実際のデータの合計長 ― 論理上のデータ長の合計 ／ 実際のデータの合計長

テーブル（表）の断片化調査 SQL 文

```
BREAK  ON  OWNER  ON  TABLE_NAME
COL  OWNER  FOR  A20
COL  TABLE_NAME  FOR  A30
COL  ALLOCATED_SIZE(MB)  FOR  990.9
COL  USED_SIZE(MB)  FOR  990.9
COL  FREE(MB)  FOR  990.9
SET  LINE  1000
SET  PAGESIZE  1000

SELECT  B.owner  "OWNER",  B.table_name  "TABLE_NAME",
        ROUND ( A.seg_bytes / 1024 / 1024 , 1 )  "ALLOCATED_SIZE(MB)",
        ROUND ( B.tab_bytes / 1024 / 1024 , 1 )  "USED_SIZE(MB)",
        ROUND ( ( A.seg_bytes - B.tab_bytes ) / 1024 / 1024 , 1 )  "FREE(MB)" ,
        ROUND (( A.seg_bytes - B.tab_bytes ) / A.seg_bytes , 1 ) * 100  断片化率
    FROM
        (
          SELECT  owner,  segment_name,  SUM ( bytes ) seg_bytes
            FROM  dba_segments  GROUP BY  owner,  segment_name
        )  A ,
        (
          SELECT  owner,  table_name,  SUM ( num_rows * avg_row_len )
        tab_bytes
            FROM  dba_tables  GROUP BY  owner,  table_name
        )  B
    WHERE  A.segment_name  =  B.table_name
      AND  B.owner  NOT IN ( 'SYS', 'SYSTEM' )
      AND  (( A.seg_bytes - B.tab_bytes ) / A.seg_bytes )  <  0.70
    ORDER BY  6 ;
```

			ALLOCATED_SIZE(MB)		
OWNER	TABLE_NAME	↓	USED__SIZE(MB)	FREE(MB)	断片化率
KOZUE	EMP	100	65	35	35

※ ７０％以下となっているテーブルは、断片化解消のメンテナンスのために、
テーブル再編成を行う

インデックス（索引）の肥大化

インデックスの階層数が４以上となっているインデックスは、肥大化解消のメンテナンスのために、インデックスの再編成が必要である

インデックス（索引）の肥大化調査 SQL 文

```
COL　OWNER　FOR　A20
COL　TABLE_NAME　FOR　A20
SET　LINE　1000
SET　PAGESIZE　1000

SELECT　owner,　index_name,　blevel　"HEIGHT"　FROM　dba_indexes
　　WHERE　blevel > 3　;
```