ORACLE

Blockchain Table関連機能強化

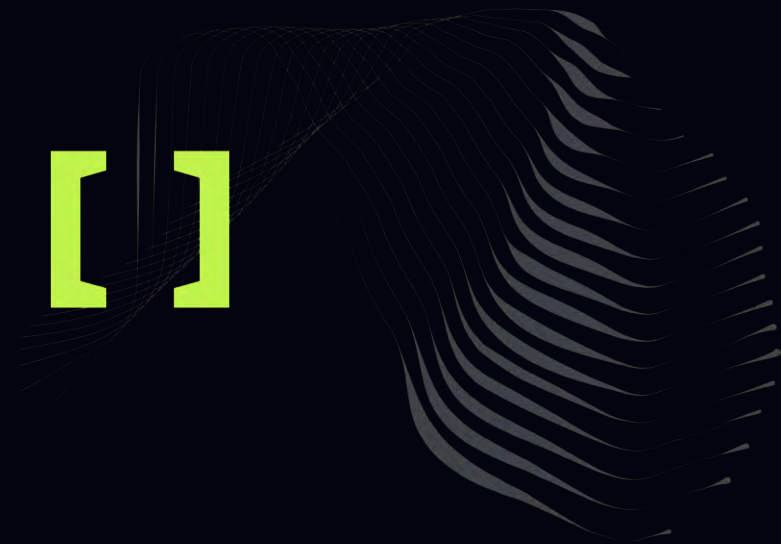
Oracle Database 23c新機能セミナー

中村　岳

日本オラクル株式会社

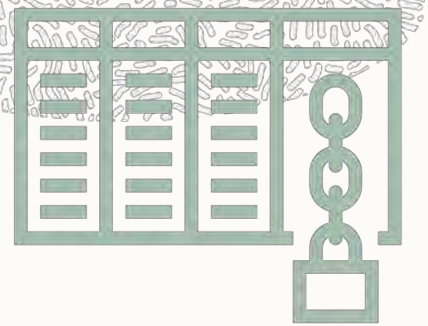
2023年10月19日

Oracle Database Blockchain Table概要



Blockchain Table：Oracle Database 21c以降&19cで利用可能（※）

耐改ざん性を追加、監査性を強化した特別なデータベース・テーブル



データベース上のレコードに耐改ざん性と監査性を付与

- 追記オンリーの不変なデータ…テーブル所有者も特権ユーザも改ざん不能
- ハッシュチェーンで行をリンク…整合性の検証、改ざんされていないことの証明が可能

Oracle Databaseの一部として高度で多彩な機能とともに容易に利用可能

- 他のテーブルと組み合わせたトランザクション
- 容易にデータ統合、多様なBIツールを用いての分析
- データベーストリガー、PL/SQLプログラムを利用したロジック表現
- レプリケーション、バックアップなどの耐障害性／高可用性機能、
  アクセスコントロールなどのセキュリティ保護機能も併用可能

※19cではRU19.11アップデートを適用することで利用可能に。
データベースの基本機能として含まれており追加ライセンスは不要（SE2でも利用可能）。

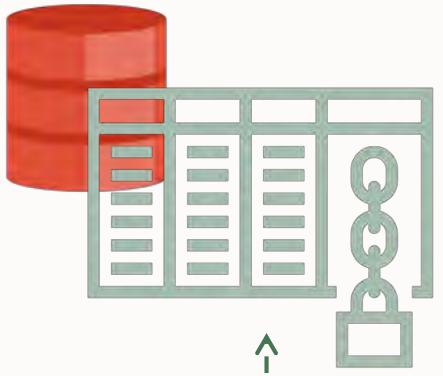
Blockchain Tableを用いることで…

情報の真正性の担保＝確実な保管と監査、証明を大幅にシンプル化

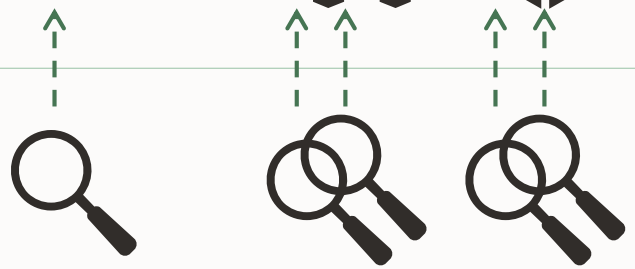
情報の確実な保管

データベースに情報を保存しつつ、
耐改ざん性のために紙原本や別媒体での記録
→**記録、保管に余分なコスト**

Blockchain Tableに保存し、
データ活用の利便性と耐改ざん性を両立



監査と証明



認証設計、アクセス制御、監査ログ、etc…
様々な情報を収集し、突き合わせて整合性を確認
複雑で手間のかかる監査プロセス

Blockchain Tableで完結した検証可能性により
監査、証明が容易

Blockchain Table: 主なユースケース

セキュリティ上重要な記録の保存

・アプリケーションのアクセスログや監査ログ
・高セキュリティエリアへの入退室記録

種々の認定、証明のエビデンス、法律上確実な保存が要求される情報

・従業員の出退勤記録、企業の会計、財務のデータ
・見積、契約、請求や支払のやり取りに係るドキュメント
・原産地証明、検査証、品質認証、RE100証明、CO2排出量証明など

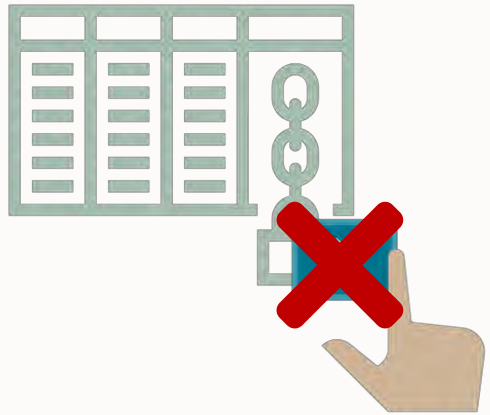
内外からの攻撃に対してデータを保護

・勘定系システム、決済系システムなどのトランザクションログ
・証券や各種アセットの所有権を管理する原簿

Blockchain Tableの特性①：データの削除、変更を制約

追記オンリーのテーブルで、イミュータブル／不変なデータを保持

- **行のDELETE**の制約（n日～無制限の保持期間を設定可能）
  - 保持期間を設定しておいた場合、INSERT後に保持期間を過ぎた行は削除することが可能
    - 通常のDELETE操作は不可で、保持期間を過ぎた行の一括削除用PL/SQLパッケージファンクションを使用
      `DBMS_BLOCKCHAIN_TABLE.delete_expired_rows()`
- **行のUPDATEとMERGE**が不可
- **テーブルのDROP**の制約（n日～無制限の保護期間を設定可能）
  - テーブルDROPは行のINSERT前なら常に可能（誤ってテーブル作成した場合すぐなら消せる）
- **テーブルのTRUNCATE**、**パーティションのDROP**が不可
- **カラムの追加／削除（※）および変更**が不可
  - 名前、データ型や一部の長さ、精度の変更、NULL制約変更が不可
  - ※カラム追加と削除は23cでの機能追加により可能になった
- Blockchain Tableの**通常のテーブルへの変換、およびその逆**の変換は不可



シンプルなCREATE BLOCKCHAIN TABLEの例

```
CREATE BLOCKCHAIN TABLE bank_ledger (bank VARCHAR2(128), deposit_date DATE,
 deposit_amount NUMBER)
        NO DROP UNTIL 31 DAYS IDLE
        NO DELETE LOCKED
        HASHING USING "SHA2_512" VERSION “V2";
```

- NO DROP UNTIL 31 DAYS IDLE
  …テーブル上の最新の行がINSERT後31日経っていないとDROPできない
- NO DELETE LOCKED
  …行は無期限にDELETEできない

Blockchain Tableの特性②：ハッシュチェーン

ハッシュ値によるデータの検証可能性により、**テーブルの中で完結した監査性**を提供

ID	User	Value	Hash
1	Tom	500	ADSJS
2	Carol	176	%SHS
3	Steve	500	SH@1
4	John	176	DHD3
5	Mike	332	*EGG
6	Sarah	632	AH11
7	Eve	25	LIO$
8	Prisha	850	SHS4

- 行のINSERT時、自動的に
  {行データ＋前の行のハッシュ値}
  に対して計算した**ハッシュ値を隠しカラムに保持**
- ある行のハッシュ値はその前の行のハッシュ値に依存し、その前の行のハッシュ値はその前の前の行のハッシュ値に依存し…
  →**ハッシュチェーンのつながりで改ざんが検知可能に**

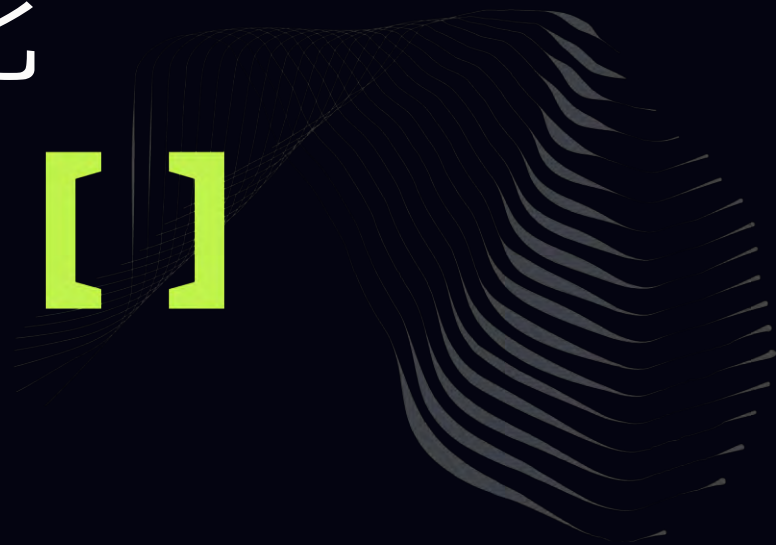
- ハッシュチェーンと行データを突合しながら辿っていくことで、**整合性（INSERT以降、行データが変更、削除されていないこと）の検証が可能**
  - 検証用PL/SQLパッケージファンクション
    DBMS_BLOCKCHAIN_TABLE.verify_rows()

Blockchain Tableの隠しカラム

作成時刻や行ハッシュ値などのデータを格納

隠しカラム	説明
INST_ID$	行が書き込まれたデータベースインスタンスを示すID
CHAIN_ID$	行が属するハッシュチェーンを示すID
SEQ_NUM$	そのハッシュチェーンの中で何番目の行かを示すシーケンス番号
CREATION_TIME$	自動的に記録される行の作成時刻
USER_NUMBER$	行を書き込んだデータベース・ユーザーのユーザー番号
HASH$	（行内容および前行のハッシュ値から計算された）ハッシュ値
SIGNATURE$	ユーザーの秘密鍵を用いて行のハッシュ値から計算されたデジタル署名（オプショナル）
SIGNATURE_ALG$	デジタル署名に使用した署名アルゴリズム
SIGNATURE_CERT$	デジタル署名に紐付いた証明書のGUID
SPARE$	現状未使用の予備カラム

Oracle Database 23cでの Blockchain Table関連の機能強化



Blockchain Table v2（バージョン2）の導入

```
CREATE BLOCKCHAIN TABLE bank_ledger (bank VARCHAR2(128), deposit_date DATE,
 deposit_amount NUMBER)
         NO DROP UNTIL 31 DAYS IDLE
         NO DELETE LOCKED
         HASHING USING "SHA2_512" VERSION “V2";
```

- **23cの機能強化のほとんどはv2のみが対象**
- 23c以降のバージョンでBlockchain Tableを作成する場合、基本的にv2を推奨
  - 既存のv1のBlockchain Tableをv2にアップグレードすることはできないので、作り直しとデータ移行が必要

Oracle Database 23cで導入されたBlockchain Table関連の機能強化

機能強化	説明	詳細
行バージョンおよび最新バージョンビュー機能の追加	行バージョンの自動採番機能、および最新バージョン行を抽出するビューの自動作成機能が追加された。	○
User Chain機能の追加	ユーザー側で定義したグループに対してハッシュチェーンを付与、検証できるようになった。	○
より柔軟なデータへの署名プロセス	代理ユーザーによる委任署名の追加など、プロセスが改善された。	○
副署プロセスの改善	副署機能が追加され行メタデータとしても格納されるようになった。	○
カラムの追加／削除が可能に	作成済のBlockchain Tableのカラムの追加、削除が可能になった。	○
Flashback Data Archiveのログの格納に対応	通常テーブルのFlashback Data Archiveの格納先にBlockchain Table を選べるようになった。	○
長期テーブル保持期間の設定に用いられる新しいユーザー権限が追加	テーブル保持期間に関するしきい値の設定が追加され、これ以上の期間を指定する場合には新たに権限（TABLE RETENTION）が要求されるようになった。	

行バージョンおよび最新バージョンビュー機能の追加

- Blockchain Table上で更新が入る情報を扱うには、
  新バージョンとしてデータを追記していくことで表現する
  - 例：ある銀行口座について、残高が更新されるたびに行を追加
- 23cでは、このような複数行によるバージョン表現の使い勝手を強化
  - テーブル作成時、最大3つのカラムを指定し、
    それらの値が同一な行を行グループとして管理
  - 例：同一のBANKおよびACCOUNT_NOの値を持つ行を
    行グループに指定
- INSERT時、自動で行グループ内のバージョン番号が採番され、
  隠しカラム（ORABCTAB_ROW_VERSION$）に格納
- さらに、それぞれの行グループから最新のバージョンの行を抽出するビュー
  （例：BANK_LEDGER_LAST$）を自動的に生成

※ROW VERSION対象のカラムの型はNUMBER、CHAR、VARCHAR2、RAWに限定

```
CREATE BLOCKCHAIN TABLE
BANK_LEDGER
    ("BANK" VARCHAR2(128 BYTE),
     "ACCOUNT_NO" NUMBER,
     "DEPOSIT_AMOUNT" NUMBER
    )
…
WITH ROW VERSION BANK_ACCOUNTS
(BANK,ACCOUNT_NO)
…
```

BANK	ACCOUNT_NO	AMOUNT	VERSION$
A BANK	123456	100	1
A BANK	123456	200	2
A BANK	**123456**	**500**	**3**

User Chain機能の追加

- あるBlockchain Table上の行のハッシュチェーンの検証には、
  そのテーブル上のすべての行へのアクセスが必要となっていた
  - 単一のテーブル上に複数種類の当事者の異なる情報が混在しているような場合に、プライバシー上の課題につながる
- 23cでは、ユーザー側で定義した行グループそれぞれに独立したハッシュチェーン（User Chain）を付与することが可能に
  ※行バージョン機能の利用が前提
- User Chainを対象としたハッシュチェーン検証機能を追加

```
CREATE BLOCKCHAIN TABLE
BANK_LEDGER
    ("BANK" VARCHAR2(128 BYTE),
     "ACCOUNT_NO" NUMBER,
     "DEPOSIT_AMOUNT" NUMBER
    )
…
WITH ROW VERSION AND USER CHAIN
BANK_ACCOUNTS (BANK,ACCOUNT_NO)
…
```

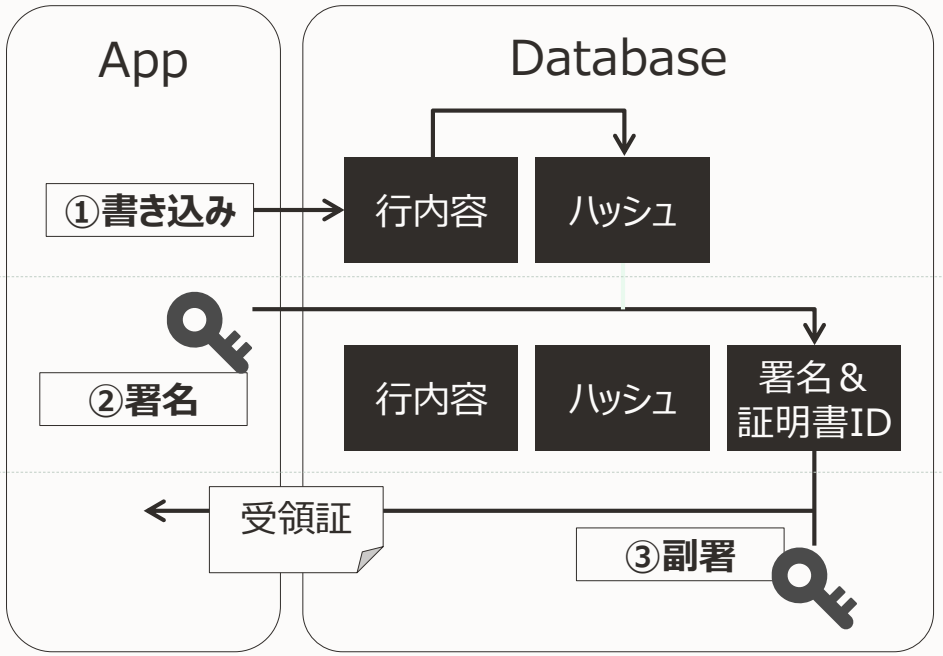
- メリット：
  - **より高速な検証**：User Chainごとに含まれる行のみ
  - **プライバシー**：検証の実行者に対しあるBlockchain Table上の行すべてではなく、あるUser Chainに属する行（同一行グループに属する行）だけを開示すれば済む

（※23c以前から存在）エンドユーザーによるデータへのデジタル署名

TRADE LEDGER

ID	User	Value	Signature	CertID
1	Tom	500	RT#E	GRTE
2	Carol	176	GI(!	SOQP
3	Wang	500	HV*P	OPRT

- Blockchain Table上の行データに対し、**エンドユーザーが自身の秘密鍵を用いて生成したデジタル署名を付与**可能
  - 秘密鍵は（DBには引き渡さず）アプリケーション側で保持し、署名の生成もDB外で実行
- エンドユーザーのデジタル証明書はDBに事前に登録
  - 証明書に照らして署名が有効かチェックされる
- **「誰が書き込んだか」を確実に区別、記録**することでセキュリティを向上し、**否認防止**にも利用可能
  - さらに応用編として、書き込まれた行内容および署名に対してテーブルオーナー側の秘密鍵を用いた副署（Countersign）の要求が可能
  - 副署を受領証としてアプリケーションに返却することで、**書き込み成功の確実性の担保**および**双方向の否認防止**が可能



署名プロセスと副署プロセスの改善

より柔軟で使いやすくなったBlockchain Table上のデータへのデジタル署名：

- INSERT実行ユーザーではない代理ユーザーによる**委任署名**（Delegate Signature）が可能に
- ユーザーカラム指定（例：主キーに指定したID系の列）での署名対象行の選択が可能に（以前は隠しカラムのinstance_id, chain_id, sequence_idの値を指定）

使いやすく改善された副署プロセス：

- 署名および委任署名を含む行データに対し、テーブルオーナーの秘密鍵により副署を生成
- 副署がリクエスト元に返却されるとともに、メタデータとしてテーブルの隠しカラムに格納されるように

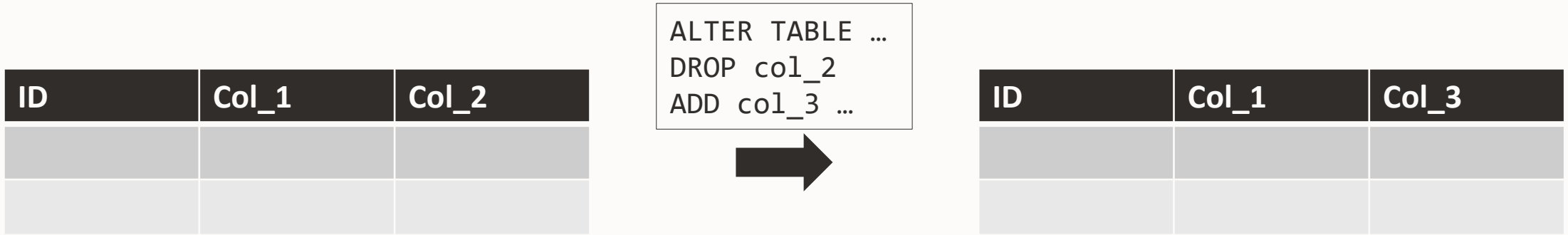
ID	User	Digest	Signature	Delegate Signature	Counter-signature
1	Alice	xxxx	Ra1H		2kxS
2	Bob	xxxx		EKoU	03Ls
3	Charlie	xxxx	b3Mx	Hs2H	93Ks

カラムの追加／削除に対応

作成済のBlockchain Tableのカラムの追加と削除（ALTER TABLE ～ ADD | DROP COLUMN）が可能に

- 業務要件の変更によりカラムの追加が必要になった、などのケースに対応
- **データにより柔軟性を求められるユースケースでもBlockchain Tableが選択肢に**

- 既存行のハッシュチェーンの検証のため、既存行のDROPされたカラムのデータは保持される
- 既存行のADDされたカラムの値はNULLになる（DEFAULT値は設定できない）



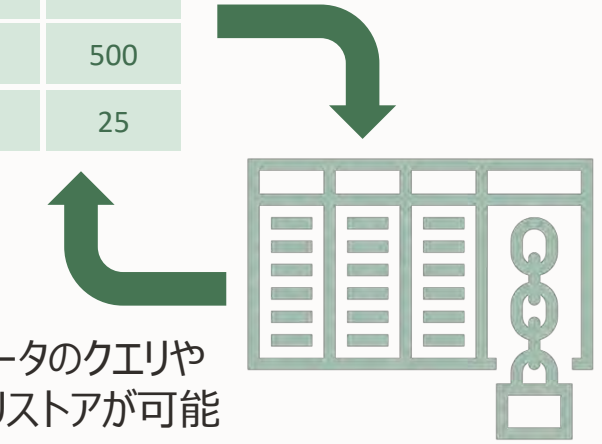
Flashback Data Archiveのログの格納に対応

- Flashback Data Archiveのログの格納先としてBlockchain Tableを指定できるようになった
- 変更履歴（UNDO情報）であるFlashback Data Archiveログを Blockchain Tableで行うことで、通常のテーブルについても変更履歴をセキュアかつ検証可能なかたちで保持できる
- テーブル作成時Flashback Data Archiveを有効化する際に、BLOCKCHAINキーワードを追加して指定することで利用が可能

```
CREATE TABLE part
(
  part_id NUMBER CONSTRAINT part_pk PRIMARY KEY,
  description VARCHAR2(50)
)
  BLOCKCHAIN FLASHBACK ARCHIVE fba_1;
```

TRADE LEDGER

ID	User	Value
1	Tom	500
2	Carol	176
3	Wang	500
4	Eve	25



Blockchain Tableの隠しカラム；v2で追加された分

隠しカラム	説明
ORABCTAB_COUNTERSIGNATURE$	テーブルオーナーの秘密鍵を用いて計算される副署のデジタル署名
ORABCTAB_COUNTERSIGNATURE_ALG$	副署に使用した署名アルゴリズム
ORABCTAB_COUNTERSIGNATURE_CERT$	副署に紐付いた証明書のGUID
ORABCTAB_COUNTERSIGNATURE_ROW_FORMAT_FLAG$	内部的にのみ使用されるカラム
ORABCTAB_COUNTERSIGNATURE_ROW_FORMAT_VERSION$	内部的にのみ使用されるカラム
ORABCTAB_DELEGATE_SIGNATURE$	代理ユーザーによる委任署名のデジタル署名
ORABCTAB_DELEGATE_SIGNATURE_ALG$	委任署名に使用した署名アルゴリズム
ORABCTAB_DELEGATE_SIGNATURE_CERT$	委任署名に紐づいた証明書のGUID
ORABCTAB_DELEGATE_USER_NUMBER$	委任署名をした代理ユーザーのユーザーID
ORABCTAB_LAST_ROW_VERSION_NUMBER$	NULLでなければ行グループの中で最新のバージョンであることを示す
ORABCTAB_PDB_GUID$	行がINSERTされたもとのPDBのGUID
ORABCTAB_ROW_VERSION$	行グループの中での行バージョン
ORABCTAB_TS$	インターバルパーティショニングがテーブルに追加された場合に用いられる仮想列
ORABCTAB_USER_CHAIN_HASH$	User Chain機能が用いられる場合にUser Chainのハッシュ値

Thank you


ORACLE