



IBM i World 2021

IBM i

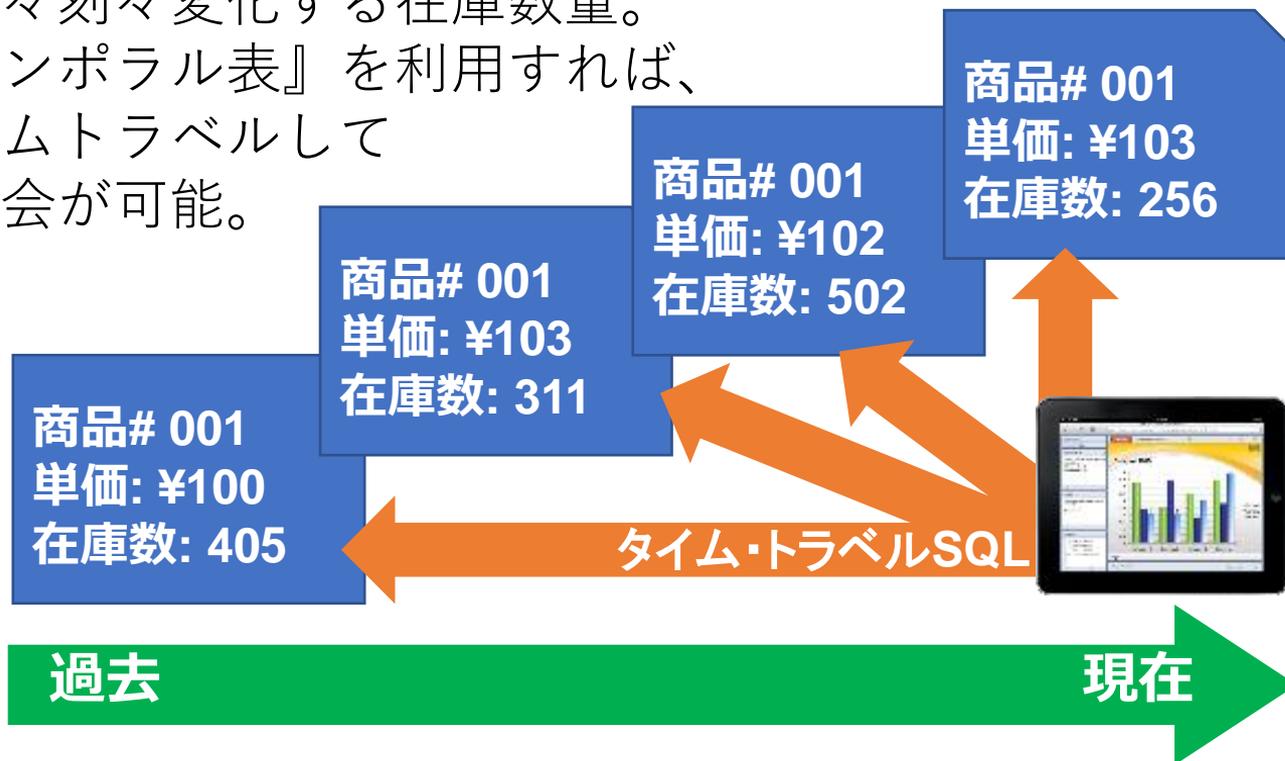
IBM i コンテンツ (12月版)

IBM i データベースの使い勝手がアップ! テンポラル表を使ってみよう

日本アイ・ビー・エム株式会社
テクノロジー事業本部
IBM Powerテクニカルセールス
澤田英寿

Db2 for i の標準機能: 業務に組み込めるタイム・トラベル照会 (『テンポラル表』機能)

例えば、時々刻々変化する在庫数量。
新機能『テンポラル表』を利用すれば、
過去にタイムトラベルして
レコード照会が可能。



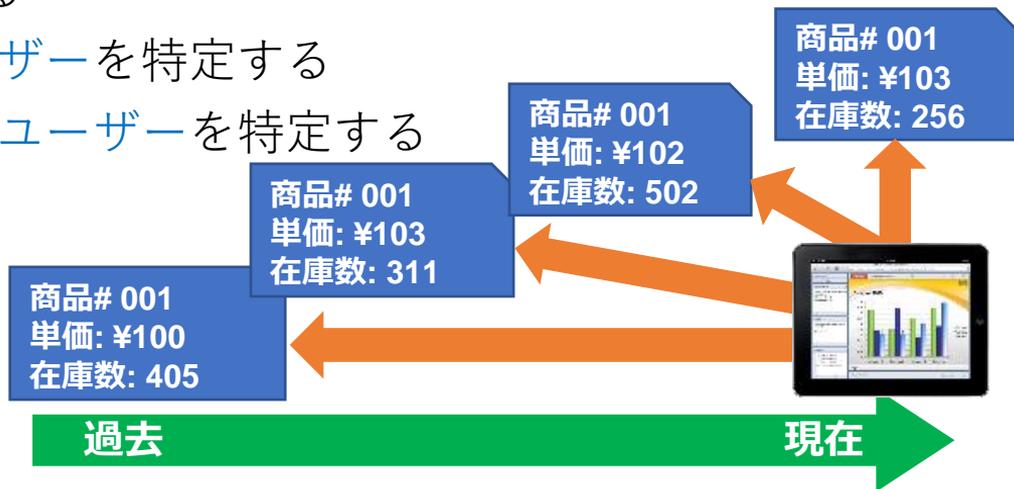
Db2 for i 「テンポラル表」 機能概要

- IBM i 7.3以降のOSの標準機能です
- 正式名は、「『システム期間』 テンポラル表」です
 - 「システム期間テンポラル表」（この資料では「テンポラル表」と記します）とは通常のデータベースの表（テーブル）に、「最終変更日時」、「終了日時」、「トランザクションID」の3つのタイム・スタンプ列と「システム期間（データがその状態であった期間）」を追加したもの
 - 上記フォーマットの行が更新・削除される都度、更新・削除される前の状態の行が「履歴表」に蓄積される。その「履歴表」により過去の時点でのデータ照会（タイムトラベル照会）が可能
- SQL言語での操作になります
 - 「システム期間テンポラル表」は、CREATE TABLE ステートメントまたは ALTER TABLE ステートメントを使用して作成できる SQL 表のこと
 - 表の履歴管理は、SQL データ操作言語 (DML) ステートメントとネイティブ DB 入出力操作の両方に対して行われます
- Db2 for iの「監査列」機能を組み合わせてセキュリティ強化にも利用可能

テンポラル表のメリットは？

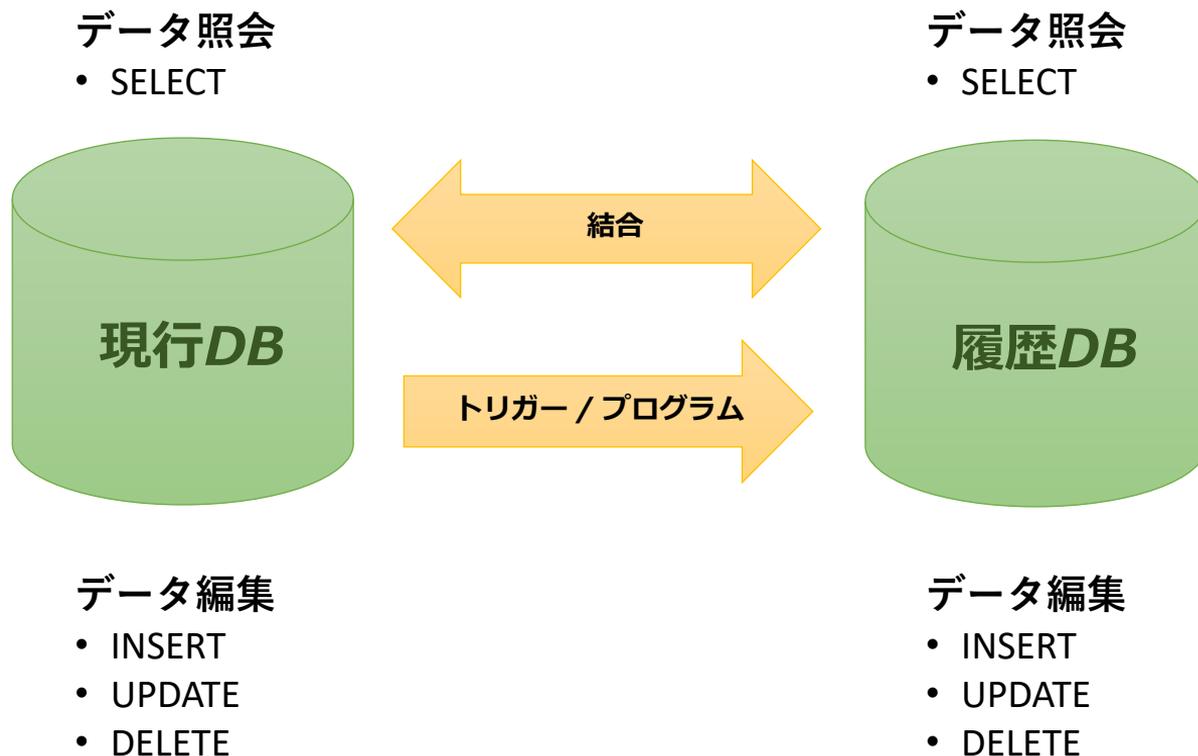
プログラムを書かなくても、履歴データ用ファイルを作らなくても、

- 日時・時間を指定して、過去の時点のレコード内容を参照できる
 - 2年前の在庫マスターに記載されていた仕入れ単価を照会する
 - 1年前間同日午前9時時点の在庫マスターに記載されていた在庫情報を照会する
 - 過去1年間の毎日18時時点の在庫数 (在庫推移) を照会する
- 監査列を利用して、監査に使える
 - レコードを削除したユーザーを特定する
 - レコードを最終更新したユーザーを特定する



IBM i 7.2以前 –データの履歴管理には手組みが必要–

- ・従来は、テープから過去データを復元して参照
- ・下記の仕組みを作るのは、複雑。夜間バッチで履歴DBを作ることが多い



IBM i 7.3 以降 – テンポラル表を使えば、システムが履歴管理 –

- ・履歴テーブルは、システムが自動で保守
- ・参照する場合も、両方のテーブルを合わせて見せてくれる

データ照会

- ・ SELECT

データ照会

- ・ SELECT



データ編集

- ・ INSERT
- ・ UPDATE
- ・ DELETE

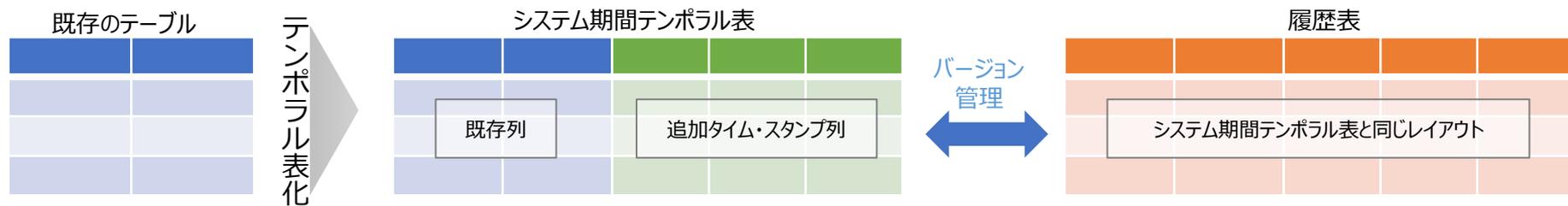
データ編集

- ・ INSERT
- ・ UPDATE
- ・ DELETE (DB管理者のみが可能)

「テンポラル表」の構成要素

【構成要素】

- 1) システム期間テンポラル表
新規もしくは既存の表に、追加で3つのタイム・スタンプ列と1つのシステム期間を追加
- 2) 履歴表
システム期間テンポラル表と同じ列を持つように定義
- 3) バージョン管理関係の構成
システム期間テンポラル表と履歴表をALTER TABLE ステートメントで接続



システム期間テンポラル表にバージョン管理が定義されると、それ以降、その表に対して更新や削除が行われると、変更前のバージョンの行が履歴表に1つの行として挿入される

よくある質問

Q: テンポラル表にすると、ストレージエリアを圧迫するのは?

A: ご心配には及びません。

テンポラル表はレコードが更新されるたびにスナップショットとして記録するので、従前より容量は増えますが、

- 1) テンポラル表に指定したファイルだけがスナップショットの対象
必要量例: $1\text{Kバイト/レコード} \times 1,000\text{回更新/日} \times 365\text{日/年} \times 10\text{年}$
 $= 3,650,000\text{Kバイト} \approx \text{約}3.5\text{Gバイト}$
- 2) IBM Power内蔵NVMeの実容量例: 3Tバイト
- 3) $3.5\text{Gバイト} \div 3\text{Tバイト} \approx \text{ストレージ容量の}1\%\text{未満}$ です。

(参考) 監査列機能の拡張

- ・表に監査列を追加
- ・行の追加・変更操作が行われると監査列の値が生成



```
ALTER TABLE TESTLIB.DEPART
ADD COLUMN audit_type_change CHAR (1)
GENERATED ALWAYS AS (DATA_CHANGE_OPERATION)
ADD COLUMN audit_user VARCHAR(128)
GENERATED ALWAYS AS (SESSION_USER)
ADD COLUMN audit_client_IP VARCHAR(128)
GENERATED ALWAYS AS (SYSIBM.CLIENT_IPADDR)
ADD COLUMN audit_job_name VARCHAR(28)
GENERATED ALWAYS AS (QSYS2.JOB_NAME)
```

テンポラル表の作成例

ここからは実際にテンポラル表を作成して、操作していきます。

まず、部門コード、部門名称、部門長コードからなる組織マスターを例に下記のデータベースを新規に作成します。

テンポラル表名：TESTLIB.DEPART

履歴表：TESTLIB.DEPART_HISTORY

とします

テンポラル表の作成例

1) テンポラル表：TESTLIB.DEPARTの作成

```
CREATE TABLE TESTLIB.DEPART  
(DEPTNO CHAR(6) NOT NULL,  
DEPTNAME CHAR(50) NOT NULL,  
MGRNO CHAR(6) NOT NULL,  
START_TS TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW BEGIN,  
END_TS TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW END,  
TS_ID TIMESTAMP(12) GENERATED ALWAYS AS TRANSACTION START ID,  
PERIOD SYSTEM_TIME(START_TS,END_TS),  
PRIMARY KEY (DEPTNO));
```

<-タイムスタンプ（開始日時）
<-タイムスタンプ（終了日時）
<-トランザクションID
<-システム期間

```
LABEL ON COLUMN TESTLIB.DEPART  
(DEPTNO IS '部門コード',  
DEPTNAME IS '部門名称',  
MGRNO IS 'MGR社員番号',  
START_TS IS 'START_TS',  
END_TS IS 'END_TS',  
TS_ID IS 'TS ID');
```

テンポラル表の作成例

2) 履歴表：TESTLIB.DEPART_HISTORY の作成

CREATE TABLE TESTLIB.DEPART_HISTORY LIKE TESTLIB.DEPART; ←テンポラル表から履歴表を作成することができる

3) バージョン管理関係を追加して、テンポラル表と履歴表の間にリンクを確立

```
ALTER TABLE TESTLIB.DEPART ADD VERSIONING USE HISTORY  
TABLE TESTLIB.DEPART_HISTORY;
```

これにより、データが更新されるとヒストリーテーブルに追加行と現在行が更新される

テンポラル表の作成例

4) サンプルデータの挿入 (下記10件)

```
INSERT INTO TESTLIB.DEPART (DEPTNO,DEPTNAME,MGRNO) VALUES('A00001','社長室','000001');
INSERT INTO TESTLIB.DEPART (DEPTNO,DEPTNAME,MGRNO) VALUES('A00010','生産本部','003302');
INSERT INTO TESTLIB.DEPART (DEPTNO,DEPTNAME,MGRNO) VALUES('A00020','開発本部','005823');
INSERT INTO TESTLIB.DEPART (DEPTNO,DEPTNAME,MGRNO) VALUES('A00030','管理本部','059478');
INSERT INTO TESTLIB.DEPART (DEPTNO,DEPTNAME,MGRNO) VALUES('A00040','営業本部','000002');
INSERT INTO TESTLIB.DEPART (DEPTNO,DEPTNAME,MGRNO) VALUES('A00011','生産管理部','000003');
INSERT INTO TESTLIB.DEPART (DEPTNO,DEPTNAME,MGRNO) VALUES('A00012','阪神工場','000004');
INSERT INTO TESTLIB.DEPART (DEPTNO,DEPTNAME,MGRNO) VALUES('A00013','中部工場','000005');
INSERT INTO TESTLIB.DEPART (DEPTNO,DEPTNAME,MGRNO) VALUES('A00014','関東工場','000006');
INSERT INTO TESTLIB.DEPART (DEPTNO,DEPTNAME,MGRNO) VALUES('A00015','和歌山工場','000007');
```

テンポラル表の作成例

5) 挿入データの照会：ここでテンポラル表と、履歴表をみてみましょう

```
SELECT * FROM TESTLIB.DEPART ORDER BY DEPTNO;
```

データの挿入時に、START_TSとEND_TSが自動で入力される

DEPTNO	DEPTNAME	MGRNO	START_TS	END_TS
A00001	社長室	000001	2021-12-13 17:18:09.796453531250	9999-12-30 00:00:00.00000000
A00010	生産本部	003302	2021-12-13 17:18:09.839346103515	9999-12-30 00:00:00.00000000
A00011	生産管理部	000003	2021-12-13 17:18:10.019338888671	9999-12-30 00:00:00.00000000
A00012	阪神工場	000004	2021-12-13 17:18:10.055366044921	9999-12-30 00:00:00.00000000
A00013	中部工場	000005	2021-12-13 17:18:10.179988371093	9999-12-30 00:00:00.00000000
A00014	関東工場	000006	2021-12-13 17:18:10.219736349609	9999-12-30 00:00:00.00000000
A00015	和歌山工場	000007	2021-12-13 17:18:10.254570478515	9999-12-30 00:00:00.00000000
A00020	開発本部	005823	2021-12-13 17:18:09.901209640625	9999-12-30 00:00:00.00000000
A00030	管理本部	059478	2021-12-13 17:18:09.939605083984	9999-12-30 00:00:00.00000000
A00040	営業本部	000002	2021-12-13 17:18:09.977620138671	9999-12-30 00:00:00.00000000

この時点の履歴表は、下記のように空です。

```
SELECT * FROM TESTLIB.DEPART_HISTORY ;
```

DEPTNO	DEPTNAME	MGRNO	START_TS	END_TS	TS_ID

テンポラル表の作成例

6) レコードの更新: レコードを更新して、テンポラル表と履歴表をみましょう

- ・部門コード (A00010)の「生産本部」を「生産技術本部」へ更新します。

```
UPDATE TESTLIB.DEPART SET DEPTNAME='生産技術本部' WHERE DEPTNO ='A00010';  
COMMIT;
```

- ・更新後のテンポラル表の照会：更新データのタイムスタンプが更新されている。

```
SELECT * FROM TESTLIB.DEPART ORDER BY DEPTNO;
```

DEPTNO	DEPTNAME	MGRNO	START_TS	END_TS
A00001	社長室	000001	2021-12-13 17:18:09.796453531250	9999-12-30 00:00:00.000000000000
A00010	生産技術本部	003302	2021-12-14 13:46:12.373277000244	9999-12-30 00:00:00.000000000000
A00011	生産管理部	000003	2021-12-13 17:18:10.019338888071	9999-12-30 00:00:00.000000000000
A00012	阪神工場	000004	2021-12-13 17:18:10.055366044921	9999-12-30 00:00:00.000000000000
A00013	中部工場	000005	2021-12-13 17:18:10.179988371093	9999-12-30 00:00:00.000000000000
A00014	関東工場	000006	2021-12-13 17:18:10.219736349609	9999-12-30 00:00:00.000000000000
A00015	和歌山工場	000007	2021-12-13 17:18:10.254570478515	9999-12-30 00:00:00.000000000000
A00020	開発本部	005823	2021-12-13 17:18:09.901209640625	9999-12-30 00:00:00.000000000000
A00030	管理本部	059478	2021-12-13 17:18:09.939605083984	9999-12-30 00:00:00.000000000000
A00040	営業本部	000002	2021-12-13 17:18:09.977620138671	9999-12-30 00:00:00.000000000000

テンポラル表の作成例

- 更新後の履歴表の照会：履歴表には更新前の古いデータが保管されている

```
SELECT * FROM TESTLIB.DEPART_HISTORY ORDER BY DEPTNO;
```

DEPTNO	DEPTNAME	MGRNO	START_TS	END_TS
A00010	生産本部	003302	2021-12-13 17:18:09.839346103515	2021-12-14 13:46:12.373277000244

テンポラル表の作成例

7) レコードの削除: レコードを削除して、テンポラル表と履歴表をみましょう

- ・ 部門コード (A00015)の「和歌山工場」を削除します。

```
DELETE TESTLIB.DEPART WHERE DEPTNO ='A00015';
COMMIT;
```

- ・ 削除のテンポラル表の照会：和歌山工場が削除されている

```
SELECT * FROM TESTLIB.DEPART ORDER BY DEPTNO;
```

DEPTNO	DEPTNAME	MGRNO	START_TS	END_TS
A00001	社長室	000001	2021-12-13 17:18:09.796453531250	9999-12-30 00:00:00.000000000000
A00010	生産技術本部	003302	2021-12-14 13:46:12.373277000244	9999-12-30 00:00:00.000000000000
A00011	生産管理部	000003	2021-12-13 17:18:10.019338888671	9999-12-30 00:00:00.000000000000
A00012	阪神工場	000004	2021-12-13 17:18:10.055366044921	9999-12-30 00:00:00.000000000000
A00013	中部工場	000005	2021-12-13 17:18:10.179988371093	9999-12-30 00:00:00.000000000000
A00014	関東工場	000006	2021-12-13 17:18:10.219736349609	9999-12-30 00:00:00.000000000000
A00020	開発本部	005823	2021-12-13 17:18:09.901209640625	9999-12-30 00:00:00.000000000000
A00030	管理本部	059478	2021-12-13 17:18:09.939605083984	9999-12-30 00:00:00.000000000000
A00040	営業本部	000002	2021-12-13 17:18:09.977620138671	9999-12-30 00:00:00.000000000000

テンポラル表の作成例

- 削除後の履歴表の照会：履歴表には削除前の古いデータが保管されている

```
SELECT * FROM TESTLIB.DEPART_HISTORY ORDER BY DEPTNO;
```

DEPTNO	DEPTNAME	MGRNO	START_TS	END_TS
A00010	生産本部	003302	2021-12-13 17:18:09.839346103515	2021-12-14 13:46:12.373277000244
A00015	和歌山工場	000007	2021-12-13 17:18:10.254570478515	2021-12-14 14:00:28.631481000244

テンポラル表の作成例

8) 過去データ（更新、削除前のデータ）を表示してみましょう

- ・更新・削除 前の日時(ここでは2021/12/14 00:00:00を指定して、過去データを照会できます。

```
SELECT * FROM TESTLIB.DEPART FOR SYSTEM_TIME AS OF '2021-12-14 00:00:00' ORDER BY DEPTNO;
```

「生産本部」と「和歌山工場」が表示されている。

DEPTNO	DEPTNAME	MGRNO	START_TS	END_TS
A00001	社長室	000001	2021-12-13 17:18:09.796453531250	9999-12-30 00:00:00.000000000000
A00010	生産本部	003302	2021-12-13 17:18:09.839346103515	2021-12-14 13:46:12.373277000244
A00011	生産管理部	000003	2021-12-13 17:18:10.019338888671	9999-12-30 00:00:00.000000000000
A00012	阪神工場	000004	2021-12-13 17:18:10.055366044921	9999-12-30 00:00:00.000000000000
A00013	中部工場	000005	2021-12-13 17:18:10.179988371093	9999-12-30 00:00:00.000000000000
A00014	関東工場	000006	2021-12-13 17:18:10.219736349609	9999-12-30 00:00:00.000000000000
A00015	和歌山工場	000007	2021-12-13 17:18:10.254570478515	2021-12-14 14:00:28.631481000244
A00020	開発本部	005823	2021-12-13 17:18:09.901209640625	9999-12-30 00:00:00.000000000000
A00030	管理本部	059478	2021-12-13 17:18:09.939605083984	9999-12-30 00:00:00.000000000000
A00040	営業本部	000002	2021-12-13 17:18:09.977620138671	9999-12-30 00:00:00.000000000000

(参考) 監査列機能の拡張

- ・表に監査列を追加
- ・行の追加・変更操作が行われると監査列の値が生成
下記のように既存のデータに、監査列を追加できます。



コマンドサンプル :

```
ALTER TABLE TESTLIB.DEPART
ADD COLUMN audit_type_change CHAR (1)
GENERATED ALWAYS AS (DATA_CHANGE_OPERATION)
ADD COLUMN audit_user VARCHAR(128)
GENERATED ALWAYS AS (SESSION_USER)
ADD COLUMN audit_client_IP VARCHAR(128)
GENERATED ALWAYS AS (SYSIBM.CLIENT_IPADDR)
ADD COLUMN audit_job_name VARCHAR(28)
GENERATED ALWAYS AS (QSYS2.JOB_NAME)
```

参考資料

- 1) IBM Systems Japan Blogより：
過去データにタイム・トラベル！簡単・便利なIBM i テンポラル表
<https://www.ibm.com/blogs/systems/jp-ja/db2-for-i-temporal-table/>
- 2) iMagazine記事より：
IBM i 7.3のテンポラル・テーブルがもたらすBI/DWHの新たな世界
<https://www.imagazine.co.jp/ibm-i-73-temporal-table/>
- 3) 監査列については下記を参照してください：
<https://www.ibm.com/docs/ja/i/7.3?topic=language-creating-auditing-columns>



ワークショップ、セッション、および資料は、IBMによって準備され、IBM独自の見解を反映したものです。それらは情報提供の目的のみで提供されており、いかなる読者に対しても法律的またはその他の指導や助言を意図したのではなく、またそのような結果を生むものでもありません。本資料に含まれている情報については、完全性と正確性を期するよう努力しましたが、「現状のまま」提供され、明示または暗示にかかわらずいかなる保証も伴わないものとします。本資料またはその他の資料の使用によって、あるいはその他の関連によって、いかなる損害が生じた場合も、IBMは責任を負わないものとします。本資料に含まれている内容は、IBMまたはそのサプライヤーやライセンス交付者からいかなる保証または表明を引き出すことを意図したもので、IBMソフトウェアの使用を規定する適用ライセンス契約の条項を変更することを意図したものでなく、またそのような結果を生むものでもありません。

本資料でIBM製品、プログラム、またはサービスに言及していても、IBMが営業活動を行っているすべての国でそれらが使用可能であることを暗示するものではありません。本資料で言及している製品リリース日付や製品機能は、市場機会またはその他の要因に基づいてIBM独自の決定権をもっていつでも変更できるものとし、いかなる方法においても将来の製品または機能が使用可能になると確約することを意図したものではありません。本資料に含まれている内容は、読者が開始する活動によって特定の販売、売上高の向上、またはその他の結果が生じると述べる、または暗示することを意図したもので、またそのような結果を生むものでもありません。パフォーマンスは、管理された環境において標準的なIBMベンチマークを使用した測定と予測に基づいています。ユーザーが経験する実際のスループットやパフォーマンスは、ユーザーのジョブ・ストリームにおけるマルチプログラミングの量、入出力構成、ストレージ構成、および処理されるワークロードなどの考慮事項を含む、数多くの要因に応じて変化します。したがって、個々のユーザーがここで述べられているものと同様の結果を得られると確約するものではありません。

記述されているすべてのお客様事例は、それらのお客様がどのようにIBM製品を使用したか、またそれらのお客様が達成した結果の実例として示されたものです。実際の環境コストおよびパフォーマンス特性は、お客様ごとに異なる場合があります。

IBM、IBM ロゴ、ibm.com、Db2、Rational、Power、POWER8、POWER9、AIXは、世界の多くの国で登録されたInternational Business Machines Corporationの商標です。

他の製品名およびサービス名等は、それぞれIBMまたは各社の商標である場合があります。

現時点での IBM の商標リストについては、www.ibm.com/legal/copytrade.shtml をご覧ください。

インテル、Intel、Intel ロゴ、Intel Inside、Intel Inside ロゴ、Centrino、Intel Centrino ロゴ、Celeron、Xeon、Intel SpeedStep、Itanium、およびPentium は Intel Corporation または子会社の米国およびその他の国における商標または登録商標です。

Linuxは、Linus Torvaldsの米国およびその他の国における登録商標です。

Microsoft、Windows、Windows NT および Windows ロゴは Microsoft Corporationの米国およびその他の国における商標です。

ITILはAXELOS Limitedの登録商標です。

UNIXはThe Open Groupの米国およびその他の国における登録商標です。

JavaおよびすべてのJava関連の商標およびロゴは Oracleやその関連会社の米国およびその他の国における商標または登録商標です。