






Power Systems  Welcome to the Waitless World 

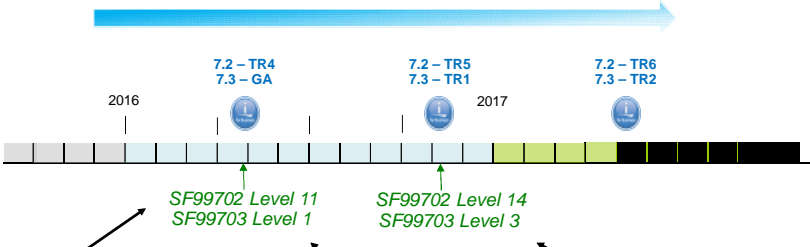
DB2 for i – Temporal Support

Rob Bestgen
 (bestgen@us.ibm.com)
 IBM - DB2 for i Consultant



Power Systems  Welcome to the Waitless World 

DB2 for i – Enhancements delivered via DB2 PTF Groups



Enhancements timed with TR4

- Inlined UDTFs
- Trigger (re)deployment
- More IBM i Services
- New DB2 built-in Global Variables
- Enhanced SQL Scalar functions
- Evaluation option for DB2 SMP & DB2 Multisystem

Enhancements in 7.3:

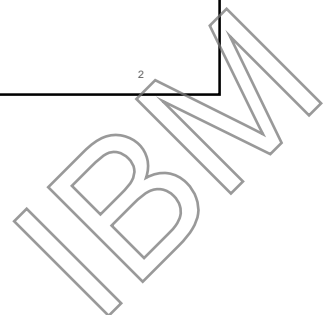
- Temporal Tables
- Generated columns for auditing
- New OLAP built-ins
- Raised architecture limits
- New support for partitioned tables
- More IBM i Services
- And much more



Enhancements timed with TR5 & TR1

- JSON_TABLE()
- INCLUDE for SQL Routines
- Database features in ACS
- Faster Scalar Functions
- More IBM i Services
- New DB2 for i Services
- And much more...

www.ibm.com/developerworks/ibmi/techupdates/db2

© 2017 IBM Corporation 2




Power Systems  Welcome to the Waitless World 



DB2 for i and IBM i 7.3 – Reasons to Upgrade

Major enhancements in DB2 for i deliver significant client value:

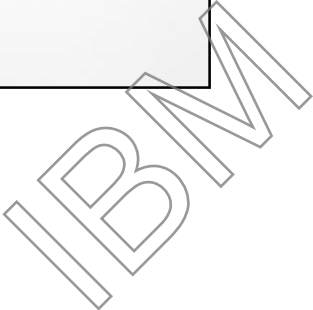
- **Temporal Tables – History of rows**
Data-centric, easily deployed, robust SQL point-in-time capability
- **Online Analytical Processing (OLAP) built-in functions**
Adding more analytics capabilities directly into DB2 for i
- **Generated Columns for auditing – Row level identity**
Let DB2 maintain the who, what, & how a row came to be
- **And... all the TR-timed enhancements delivered to IBM i 7.2**
IBM i Services, VARCHAR_FORMAT, built-in global variables, and many more





© 2017 IBM Corporation 3

Power Systems  Welcome to the Waitless World 

Temporal



2


Power Systems  Welcome to the Waitless World 

What is Temporal?

definition (from dictionary.com)



Temporal:
adjective

- of or relating to time
- enduring for a time only; temporary; transitory



From a database perspective, 'temporal' means managing and maintaining time-related data - **rows in a table**
(the temporary part is up to you 😊)


© 2017 IBM Corporation 5

Power Systems  Welcome to the Waitless World 

What is DB2 Temporal support?

To **support** temporal data, database **must** satisfy multiple requirements

- Augment data to provide a time dimension
- Retain/safeguard history of data changes over time
- Provide a way to identify when data changes occur and when data was/is relevant
- Ensure 'normal' access uses the latest data and is not 'confused' with history

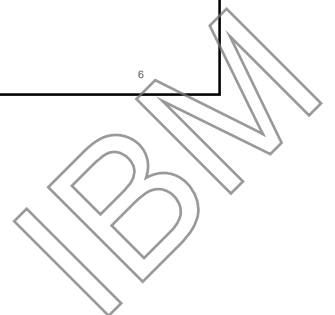



Example of data to be tracked:

- a) Savings account is opened on Monday
- b) Deposit is made mid Tuesday
- c) Withdrawals are made Wednesday and Thursday
- d) Deposits and withdrawals made Friday


Following Monday – banking officials investigate for possible money laundering (joke)

© 2017 IBM Corporation 6






Power Systems

Welcome to the Waitless World 

What is DB2 Temporal support cont...

To simplify **usage** of temporal data, database **should** provide certain features

- Provide a reasonable way to view the data from a certain point in time
- Simplify the work to apply a point-in-time view for a set of operations e.g. across an application
- Potentially support a way to view data for a specific span of time




Examples:

- What was the account balance Tuesday morning?
- How many deposits and withdrawals occurred between Tuesday 10AM and Thursday 4:30PM?
- Compare the account balance of Monday evening vs. Thursday morning


Auditors love this stuff

© 2017 IBM Corporation

7




Power Systems

Welcome to the Waitless World 

More temporal examples

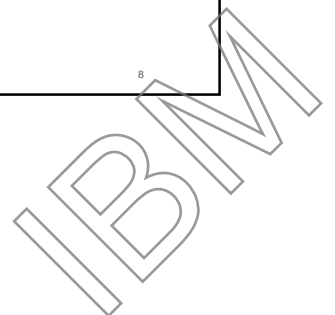
With Temporal Tables, you can answer time-based questions:



- Who was the client rep two years ago?
- Who were the client reps over the last five years?
- Produce an inventory report using a different point in time





© 2017 IBM Corporation

8



Power Systems  Welcome to the Waitless World 

DB2 for i & Temporal

Power Systems  Welcome to the Waitless World 

IBM i – Common ‘Do It Yourself’ options

Option 1: journals and scraping

Accessing Data
• SELECT

table

Modifying Data
• INSERT
• UPDATE
• DELETE

Changes

Accessing Data
• Specialized reader

Journal

**Tedious
Specialized
Limited**

Option 2: history table

Accessing Data
• SELECT

Current table

UNION

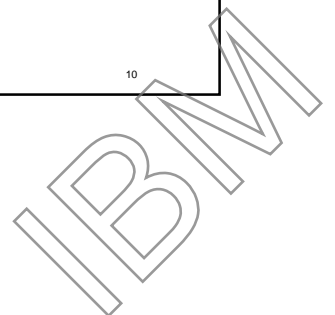
Triggers

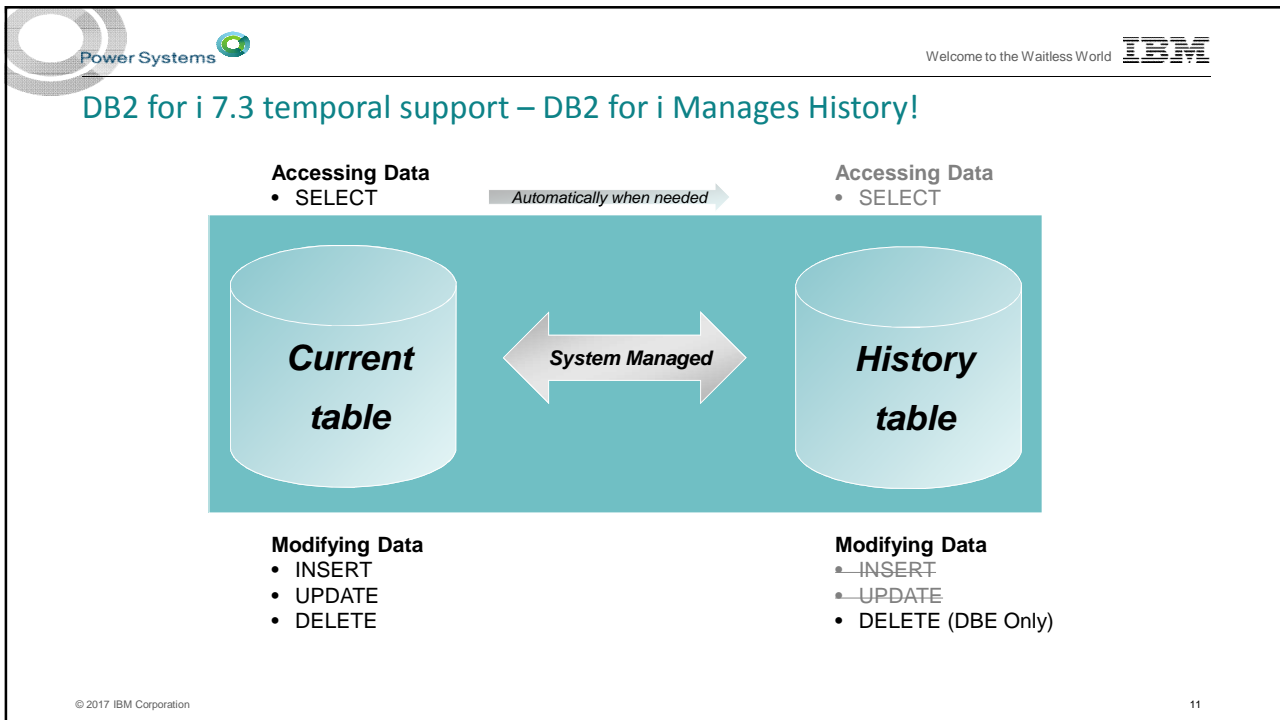
Accessing Data
• SELECT



History table

Modifying Data
• INSERT
• UPDATE
• DELETE

© 2017 IBM Corporation 10






Power Systems  Welcome to the Waitless World 

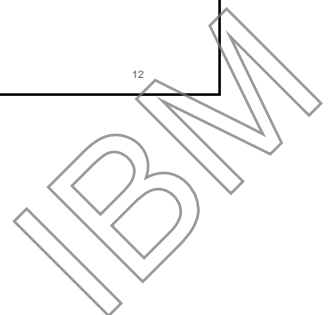
DB2 for i and Temporal Tables

With DB2 Temporal Tables, you can ask:

- **Who was the client rep two years ago?**
SELECT CLIENT_REP FROM ACCOUNTS
FOR SYSTEM_TIME AS OF CURRENT_TIMESTAMP – 2 YEARS
- **Who were the client reps over the last five years?**
SELECT CLIENT_REP FROM ACCOUNTS
FOR SYSTEM_TIME FROM CURRENT_TIMESTAMP – 5 YEARS
TO CURRENT_TIMESTAMP
- **Run the inventory report using a different point in time**
SET **CURRENT TEMPORAL SYSTEM_TIME** '2016-12-26 17:00:00';
CALL GENERATE_INVENTORY_REPORT();



© 2017 IBM Corporation 12



How? The foundation

DB2 keeps the history of each row in a temporal table over time

- **Q:** Is this the same as keeping history of a **business** transaction?



What pieces are needed for tracking data (row) history?


- A repository for the rows' history – **history table**
- An identifier to track a row as it changes over time – **transaction id**
- When was a row's historic value valid? – **row begin and row end** timestamps


Configuring a Temporal Table

```
ALTER TABLE employee
  ADD COLUMN instance_begin
    TIMESTAMP(12) NOT NULL
    GENERATED ALWAYS AS ROW BEGIN
  ADD COLUMN instance_end
    TIMESTAMP(12) NOT NULL
    GENERATED ALWAYS AS ROW END
  ADD COLUMN transaction_id
    TIMESTAMP(12)
    GENERATED ALWAYS AS TRANSACTION START ID ADD PERIOD
    SYSTEM_TIME (instance_begin, instance_end) Establish birth/death of a row
```

```
CREATE TABLE employee_history LIKE employee Create history table
```

```
ALTER TABLE employee ADD VERSIONING USE HISTORY TABLE
employee_history Enable Temporal tracking
```



Welcome to the Waitless World 

Configuring a Temporal Table

```

ALTER TABLE employee
  ADD COLUMN instance_begin
    TIMESTAMP(12) NOT NULL IMPLICITLY HIDDEN
    GENERATED ALWAYS AS ROW BEGIN
  ADD COLUMN instance_end
    TIMESTAMP(12) NOT NULL IMPLICITLY HIDDEN
    GENERATED ALWAYS AS ROW END
  ADD COLUMN transaction_id
    TIMESTAMP(12) IMPLICITLY HIDDEN
    GENERATED ALWAYS AS TRANSACTION START ID ADD PERIOD
    SYSTEM_TIME (instance_begin, instance_end)

```

Establish birth/death of a row

```

CREATE TABLE employee_history LIKE employee

```

Create history table


```


ALTER TABLE employee ADD VERSIONING USE HISTORY TABLE
employee_history

```

Enable Temporal tracking

© 2017 IBM Corporation 15

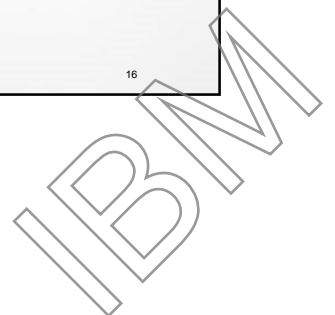



Welcome to the Waitless World 

Temporal FAQs


- Can there be more than one temporal table on my system?
 - Absolutely. Temporal is per table, so many tables can be made temporal (except history tables of course)
- Can a DDS file (not just DDL/SQL) be made temporal?
 - Yes, though the three timestamp fields must exist in the file, which requires ALTER TABLE to add
- Is journaling required? -- Yes
- Can I mix different FOR SYSTEM_TIME settings in different parts of a select e.g. two sides of a UNION can be at different SYSTEM_TIMES? -- Yes
- Are constraints supported?
 - Yes on the main table, but constraints are not allowed on the history table

16






Power Systems

Welcome to the Waitless World 


Temporal FAQs...

- Can I attach an existing table as the history table?
 - Yes, but columns and attributes must exactly match. Simplest is to use CREATE TABLE LIKE
- Can the history table have a superset of columns of the main table? -- No
- Can columns in the history table be in a different order than the corresponding columns in the main table? -- No
- Can column attributes be different (but compatible) between main table and history table?
 - No, attributes must exactly match
- Can I ALTER TABLE a temporal (main) table to add a column?
 - Yes. The database automatically adds the same column to the history table
- Can I ALTER TABLE of a temporal (main) table to drop a column?
 - No. History could be lost so database prevents it

17



Power Systems

Welcome to the Waitless World 

Accessing a Temporal Table

SQL access is by far preferred for **time-based access** since ease of access is built into the language

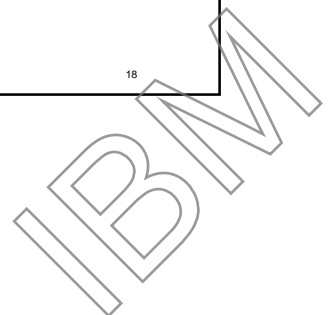
- When SQL statements reference the current table, DB2 for i automatically accesses the history table as needed
- New clauses on the SELECT statement
 - FOR SYSTEM_TIME AS OF <value>
 - FOR SYSTEM_TIME FROM <value> TO <value>
 - FOR SYSTEM_TIME BETWEEN <value> AND <value>
- New special register
 - CURRENT TEMPORAL SYSTEM_TIME


Note:

- Native access is manual work. Current and history tables are considered different accesses
- Database **does** still manage (and enforce) history tracking, even for native driven data changes


© 2017 IBM Corporation

18





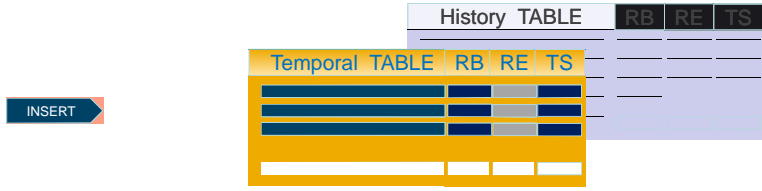
Power Systems

Welcome to the Waitless World 

Temporal in motion


Inserting rows does not impact the history table

- ROW BEGIN (RB) Column – timestamp when the row was born
- ROW END (RE) Column – set to “end of time”




© 2017 IBM Corporation

19



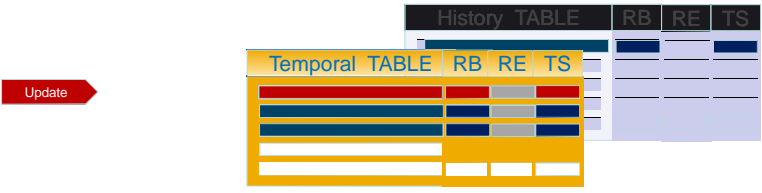
Power Systems

Welcome to the Waitless World 

Temporal in motion

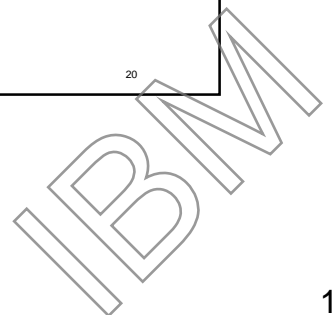
Updating rows causes rows to be added to the history table



- ROW BEGIN (RB) Column – timestamp when the row was born
- ROW END (RE) Column – the death of the row results in the RE of the historical row matching the RB of the active row



© 2017 IBM Corporation

20

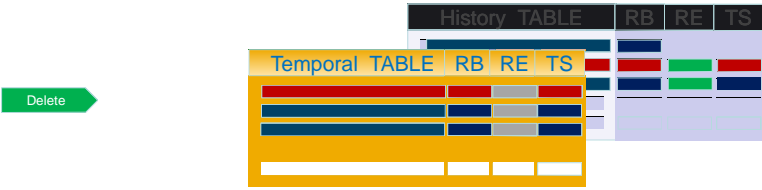


Power Systems  Welcome to the Waitless World 



Temporal in motion

Deleting rows removes them from the temporal table and adds them to history table

- ROW END (RE) Column – set to the death time of the row



© 2017 IBM Corporation 21

Power Systems  Welcome to the Waitless World 

Optional autogenerated columns

DATA CHANGE OPERATION

- one character value recording the last data change:
 - I = Insert
 - U = Update
 - D = Delete

Note: the Delete record will be included if the temporal table was configured with the ON DELETE ADD EXTRA ROW clause.


SESSION_USER

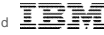
- (var)char containing the user profile currently in use which identifies who is making the data change to the database

Ex:

```
ALTER TABLE fact_table
  ADD COLUMN audit_type_change CHAR (1)
  GENERATED ALWAYS AS (DATA CHANGE OPERATION)
  ADD COLUMN audit_user VARCHAR(128)
  GENERATED ALWAYS AS (SESSION_USER)
```

© 2017 IBM Corporation 22



Welcome to the Waitless World 

Data Change Operation and Row-level Auditing detail

History table stores previous versions of a system-period temporal table's rows

- ROW BEGIN (RB) Column – timestamp when the rows were born
- ROW END (RE) Column – set to “end of time”
- Data Change Operation (CHG) – ‘I’ for INSERT
- Session User (USR) – identity of inserter


Insert


Temporal TABLE	RB	RE	TS	CHG	USR
				I	Tom
				I	Tom
				I	Tom

History TABLE	RB	RE	TS	CHG	USR

© 2017 IBM Corporation

23



Welcome to the Waitless World 

Data Change Operation and Row-level Auditing detail

History table stores previous versions of a system-period temporal table's rows

- ROW BEGIN (RB) Column – Birth
- ROW END (RE) Column – Death
- Data Change Operation (CHG) – ‘U’ for UPDATE
- Session User (USR) – identity of updater

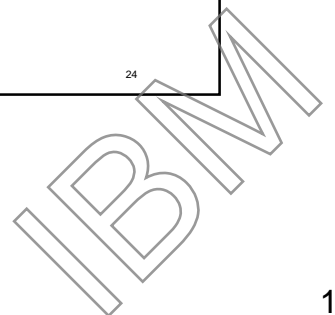
Update


Temporal TABLE	RB	RE	TS	CHG	USR
				U	Nick
				I	Tom
				I	Tom


History TABLE	RB	RE	TS	CHG	USR

© 2017 IBM Corporation

24





Welcome to the Waitless World 

ON DELETE ADD EXTRA ROW – in motion

History table stores previous versions of a system-period temporal table's rows

- ROW BEGIN (RB) Column – Birth
- ROW END (RE) Column – Death
- Data Change Operation (CHG) – 'D' for DELETE
- Session User (USR) – identity of deleter


Delete


Temporal TABLE	RB	RE	TS	CHG	USR
				U	Nick
				I	Tom
				I	Tom

History TABLE	RB	RE	TS	CHG	USR
					Tom
					Nick
					Jim
					Tom
					Jim

© 2017 IBM Corporation

25



Welcome to the Waitless World 

Temporal – more example queries

- Compare balances **between** different points in time for account 88880001

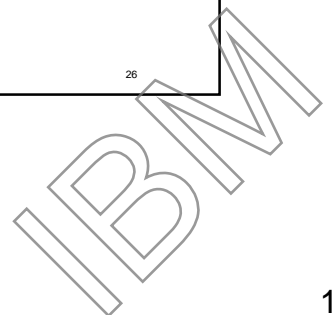
```



SELECT T1.BALANCE AS BALANCE_2013,
       T2.BALANCE AS BALANCE_2014
FROM account FOR SYSTEM_TIME AS OF '2013-12-31' T1,
     account FOR SYSTEM_TIME AS OF '2014-12-31' T2
WHERE T1.ACCT_ID = '88880001' AND T2.ACCT_ID = '88880001';
    
```

BALANCE_2013	BALANCE_2014
50000.00	60000.00

© 2017 IBM Corporation

26



Power Systems  Welcome to the Waitless World 

Temporal – more example queries



- Query all versions of rows for account '88880001'

```
SELECT ACCT_ID,
       BALANCE,
       BALANCE - LAG(BALANCE,1,0)
       OVER(ORDER BY TRANSACTION_TIME) AS CHANGES,
       TRANSACTION_TIME,
       ROW_DEATH
FROM account FOR SYSTEM_TIME
BETWEEN '0001-01-01' AND '9999-12-30'
WHERE ACCT_ID= '88880001'
ORDER BY TRANSACTION_TIME ASC;
```

LAG is one of many new OLAP specifications added in IBM i 7.3

ACCT_ID	BALANCE	CHANGES	TRANSACTION_TIME	INSTANCE_END
88880001	3000.00	-2990.00	2013-01-02 10:02:16.987139000000	2013-05-05 14:36:16.637149000000
88880001	10.00	49990.00	2013-05-05 14:36:16.637149000000	2013-12-30 10:50:59.637124000000
88880001	50000.00	-41000.00	2013-12-30 10:50:59.637124000000	2014-01-05 10:50:59.611224000000
88880001	9000.00	-8000.00	2014-01-05 10:50:59.611224000000	2014-03-05 21:12:23.321216000000
88880001	1000.00	-900.00	2014-03-05 21:12:23.321216000000	2014-09-01 14:01:11.111231000000
88880001	100.00	59900.00	2014-09-01 14:01:11.111231000000	2014-12-20 10:05:18.617454000000
88880001	60000.00	-60000.00	2014-12-20 10:05:18.617454000000	9999-12-30 00:00:00.000000000000

© 2017 IBM Corporation 27

Power Systems  Welcome to the Waitless World 

SYSTEM - Bind Option



System Time Sensitivity is controlled at the program level:

- SYSTEM_TIME_SENSITIVE column within catalog QSYS2.SYSPROGRAMSTAT
 - NULL or 'NO' – Program is not time sensitive
 - 'YES' – Program is time sensitive
- Programs built prior to IBM i 7.3 are by default, **not time sensitive**
 - CURRENT TEMPORAL SYSTEM_TIME is ignored
- Programs (re)built on IBM i 7.3 are by default, **time sensitive**
 - CURRENT TEMPORAL SYSTEM_TIME is applied when queries reference temporal tables

Build time controls:

- Routines (SQL/External) → SET OPTION SYSTIME = *YES or *NO
- CRTSQLxxx → OPTION(*SYSTIME or *NOSYSTIME)
- RUNSQLSTM → SYSTIME(*YES or *NO)

© 2017 IBM Corporation 28

Power Systems  Welcome to the Waitless World 

Considerations with temporal tables

Aggregation (GROUP BY)

- Be very careful aggregating across time ranges: **FOR SYSTEM_TIME BETWEEN** or **FOR SYSTEM TIME FROM**. A 'row' could be counted several times!


Ex:

```
SELECT COUNT(*) FROM ORDER_HEADER  
FOR SYSTEM_TIME BETWEEN '0001-01-01' AND '9999-12-30' ;
```



Does NOT count the total number of orders since the beginning!

Joins (joining tables)

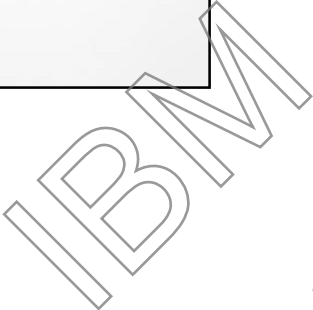
- From a business perspective, do all tables need to be temporal?
 - Is a subset of tables enabled as temporal enough?
- Could the join column(s) change when a row is updated?
 - If so, think through update situations to ensure answers are consistent




© 2017 IBM Corporation 29


Power Systems  Welcome to the Waitless World 

DB2 for i & Row Level Auditing






Power Systems

Welcome to the Waitless World 


Enhanced data-centric auditing – with autogenerated columns

- **Autogenerated columns are a very powerful building block for datacentric programming in that they direct the database to automatically generate column values**
 - Database manages them. Row values cannot be altered, even by a developer
- **Prior to IBM i 7.3, DB2 for i supported:**
 - IDENTITY columns (which are very good for surrogate primary keys)
 - ROW CHANGE TIMESTAMP (which records the time whenever a row is changed)
- **The SQL syntax GENERATED ALWAYS prevents anyone from modifying those column values, including a knowledgeable hacker**
- **IBM i 7.3 includes support for additional options:**
 - DATA CHANGE OPERATION (I/U/D)
 - Special register
 - Built-in Global Variable

© 2017 IBM Corporation 31



Power Systems

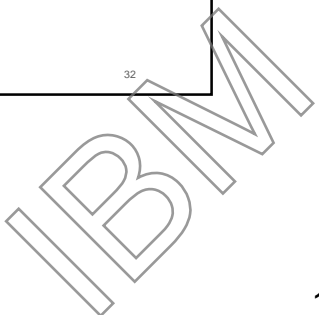
Welcome to the Waitless World 



Autogenerated columns – DATA CHANGE OPERATION

- **DATA CHANGE OPERATION is a one character value recording the last data change:**
 - I = Insert
 - U = Update
 - D = Delete
- **These work well with temporal tables in that history table will provide a timeline of what changes were made and when**
 - **The Delete record will be included if the temporal table was configured with the ON DELETE ADD EXTRA ROW clause**

```
ALTER TABLE fact_table
  ADD COLUMN audit_type_change CHAR (1)
  GENERATED ALWAYS AS (DATA CHANGE OPERATION)
```

© 2017 IBM Corporation 32




Power Systems  Welcome to the Waitless World 

Special Registers and Global Variables (Review)



DB2 provides different ways to communicate across an application flow. Two are of particular interest:

- 1. Special Registers**
 - Predefined special values that can be referenced in SQL
Examples: CURRENT USER, CURRENT TIMESTAMP, CURRENT DATE...
 - Most registers are maintained by the database. However, some registers can be SET by the application
Examples: CLIENT_ACCTNG, CLIENT_USERID...
- 2. Global Variables**
 - Variables that can be created and used across SQL statements
Example:

```
CREATE VARIABLE QGPL.MYVAR INT DEFAULT 123
...
SELECT * FROM MYTAB WHERE MYCOL = QGPL.MYVAR
```
 - Database defines and manages some built-in global variables
Examples: QSYS2.JOB_NAME, SYSIBM.CLIENT_IPADDR



© 2017 IBM Corporation 33

Power Systems  Welcome to the Waitless World 


Autogenerated columns – special registers


- **Special registers can be used to record information about the user making the change and/or the application environment**
- **Client registers can be set by the application to provide additional application information**
- **CURRENT SERVER contains the currently connected server**
- **SESSION_USER and USER contain the user profile currently in use which identifies who is making a change to the database**

CURRENT CLIENT_ACCTNG
CURRENT CLIENT_APPLNAME
CURRENT CLIENT_PROGRAMID
CURRENT CLIENT_USERID
CURRENT CLIENT_WRKSTNNAME
CURRENT SERVER
SESSION_USER
USER

```
ALTER TABLE fact_table
  ADD COLUMN audit_app_client_userid VARCHAR(255)
  GENERATED ALWAYS AS (CURRENT CLIENT_USERID)
  ADD COLUMN audit_user VARCHAR(128)
  GENERATED ALWAYS AS (SESSION_USER)
```

© 2017 IBM Corporation 34



Welcome to the Waitless World 


Autogenerated columns – built-in global variables


- **Built-in global variables are managed by the system and provide additional environmental information**
- **You can use these to monitor things like which job or which IP address is being used to make a change to the database**

```
ALTER TABLE fact_table
  ADD COLUMN audit_job_name VARCHAR(28)
    GENERATED ALWAYS AS (QSYS2.JOB_NAME)
  ADD COLUMN audit_client_IP VARCHAR(128)
    GENERATED ALWAYS AS (SYSIBM.CLIENT_IPADDR)
```


QSYS2.JOB_NAME
QSYS2.SERVER_MODE_JOB_NAME
SYSIBM.CLIENT_HOST
SYSIBM.CLIENT_IPADDR
SYSIBM.CLIENT_PORT
SYSIBM.PACKAGE_NAME
SYSIBM.PACKAGE_SCHEMA
SYSIBM.PACKAGE_VERSION
SYSIBM.ROUTINE_SCHEMA
SYSIBM.ROUTINE_SPECIFIC_NAME
SYSIBM.ROUTINE_TYPE



© 2017 IBM Corporation
35



Welcome to the Waitless World 

More Information



Power Systems  Welcome to the Waitless World 

DB2 for IBM i Resources

- **DB2 for IBM i homepage:** www.ibm.com/systems/power/software/i/db2

IT infrastructure > Power Systems > Software > IBM i >


IBM DB2 for i

Overview Benefits Getting started Products Resources

DB2 for i (formerly known as DB2 for i5/OS) is an advanced, 64-bit Relational Database Management System (RDBMS) that leverages the high performance, virtualization, and energy efficiency of the IBM i environment.

You are in: [DB2 for i Wiki](#) > Welcome to DB2 for i

Welcome to DB2 for i



 | Updated yesterday at 7:18 PM by [drmack2](#) | Tags: None

Page Actions ▾

Welcome to the home page for the DB2 for i Wiki. Here you will find a variety of information from the leading experts for DB2 for i within IBM.

- **DB2 for IBM i wiki:**
ibm.biz/Bd4fFb

© 2017 IBM Corporation 37

Power Systems  Welcome to the Waitless World 

DB2 for IBM i Lab Services

- **Facilitated workshops covering current state, requirements, future state, possible solutions, implementation best practices, and formulation of a strategic roadmap:**
 - RCAC
 - **Temporal Tables**
- **Customized consulting workshops**
 - **Advanced SQL and Datacentric Programming**
 - **SQL Performance Best Practices, Monitoring and Tuning**
- **Consulting on any DB2 for i topic**

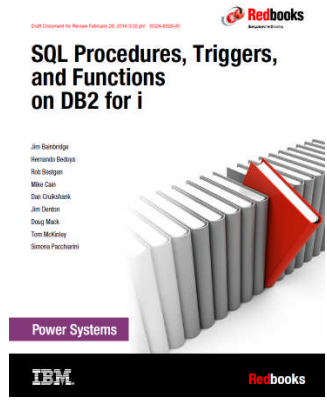
For more information, contact mcaain@us.ibm.com

© 2017 IBM Corporation 38



DB2 for i – SQL Programming Resources

**Essential resource for SQL & DB2
for i database application
development**
Refreshed in 2016

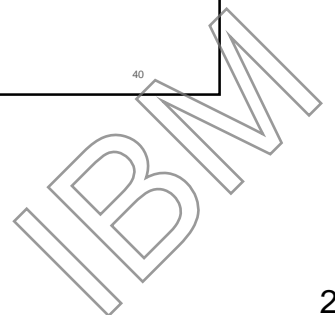


www.redbooks.ibm.com/redpieces/abstracts/sg248326.html



Thank You!

www.ibm.com/developerworks/ibmi/techupdates/db2





Special notices

This document was developed for IBM offerings in the United States as of the date of publication. IBM may not make these offerings available in other countries, and the information is subject to change without notice. Consult your local IBM business contact for information on the IBM offerings available in your area.

Information in this document concerning non-IBM products was obtained from the suppliers of these products or other public sources. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. Send license inquiries, in writing, to IBM Director of Licensing, IBM Corporation, New Castle Drive, Armonk, NY 10504-1785 USA.

All statements regarding IBM future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

The information contained in this document has not been submitted to any formal IBM test and is provided "AS IS" with no warranties or guarantees either expressed or implied.

All examples cited or described in this document are presented as illustrations of the manner in which some IBM products can be used and the results that may be achieved. Actual environmental costs and performance characteristics will vary depending on individual client configurations and conditions.

IBM Global Financing offerings are provided through IBM Credit Corporation in the United States and other IBM subsidiaries and divisions worldwide to qualified commercial and government clients. Rates are based on a client's credit rating, financing terms, offering type, equipment type and options, and may vary by country. Other restrictions may apply. Rates and offerings are subject to change, extension or withdrawal without notice.

IBM is not responsible for printing errors in this document that result in pricing or information inaccuracies.

All prices shown are IBM's United States suggested list prices and are subject to change without notice; reseller prices may vary.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

Any performance data contained in this document was determined in a controlled environment. Actual results may vary significantly and are dependent on many factors including system hardware configuration and software design and configuration. Some measurements quoted in this document may have been made on development-level systems. There is no guarantee these measurements will be the same on generally-available systems. Some measurements quoted in this document may have been estimated through extrapolation. Users of this document should verify the applicable data for their specific environment.

Revised September 26, 2006