DB2 Migration Planning Workshop

IBM

IBM Software Group

DB2 10 for z/OS Security Overview

DB2 Information Management Software

Disclaimer and Trademarks

Information contained in this material has not been submitted to any formal IBM review and is distributed on "as is" basis without any warranty either expressed or implied. Measurements data have been obtained in laboratory environment. Information in this presentation about IBM's future plans reflect current thinking and is subject to change at IBM's business discretion. You should not rely on such information to make business plans. The use of this information is a customer responsibility.

IBM MAY HAVE PATENTS OR PENDING PATENT APPLICATIONS COVERING SUBJECT MATTER IN THIS DOCUMENT. THE FURNISHING OF THIS DOCUMENT DOES NOT IMPLY GIVING LICENSE TO THESE PATENTS.

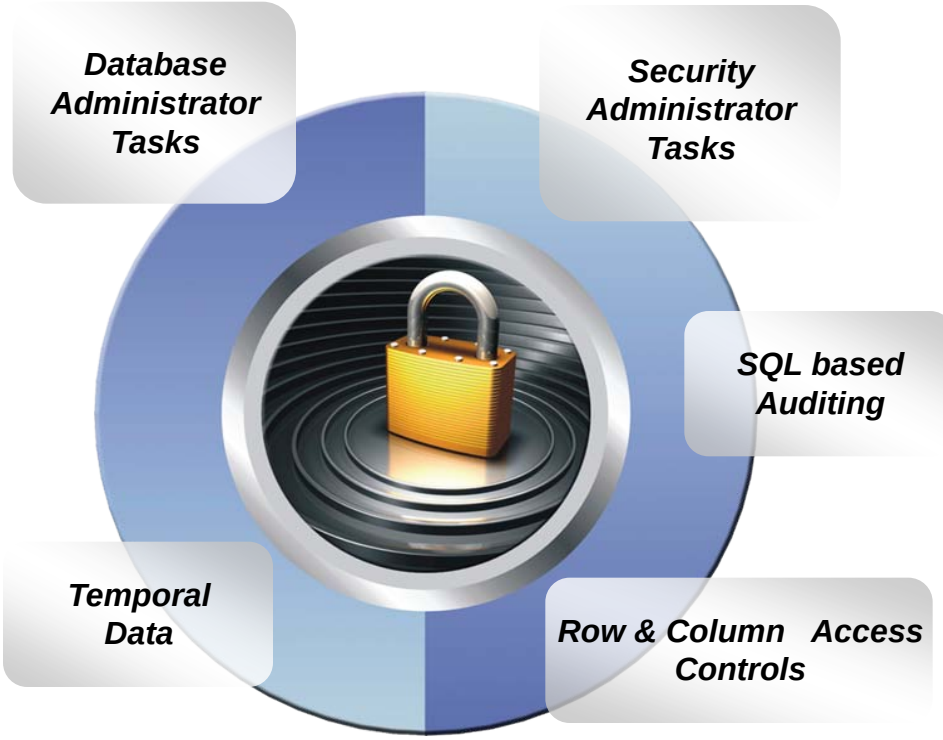
TRADEMARKS: THE FOLLOWING TERMS ARE TRADEMARKS OR ® REGISTERED TRADEMARKS OF THE IBM CORPORATION IN THE UNITED STATES AND/OR OTHER COUNTRIES: AIX, DATABASE 2, DB2, Enterprise Storage Server, FICON, FlashCopy, Netfinity, RISC, RISC SYSTEM/6000, System i, System p, System x, System z, IBM, Lotus, NOTES, WebSphere, z/Architecture, z/OS, zSeries

The FOLLOWING TERMS ARE TRADEMARKS OR REGISTERED TRADEMARKS OF THE MICROSOFT CORPORATION IN THE UNITED STATES AND/OR OTHER COUNTRIES: MICROSOFT, WINDOWS, WINDOWS NT, ODBC, WINDOWS 95

For additional information see ibm.com/legal/copytrade.phtml

Satisfy Auditors: New DB2 data-centric features

- **Improved data protection**
  - Minimize the use of a superuser authorities such as SYSADM
  - A different group should manage access to restricted data than the owner of the data
- **Improved data privacy**
  - All dynamic access to tables containing restricted data needs to be protected
- **Improved data auditing**
  - Any dynamic access or use of a privileged authority needs to be included in your audit trail
  - Maintain historical versions of data for years or during a business period



Today's Mainframe:
The power of industry-leading security, the simplicity of centralised management

Reduce risk by minimizing use of SYSADM

New granular system authorities

- **Prior to DB2 10**
- **SYSADM**
- **DBADM**
- **DBCTRL**
- **DBMAINT**
- **SYSCTRL**
- **PACKADM**
- **SYSOPR**

New in DB2 10

- **System DBADM**
  - **ACCESSCTRL**
  - **DATAACCESS**
- **SECADM**
- **SQLADM**
- **EXPLAIN**

SECADM authority for performing security tasks

- **SECADM authority**
  - Separates security duties from SYSADM and SYSCTRL
  - Prevents SYSADM and SYSCTRL from granting or revoking privileges
  - Can not access or change the data
- **What can SECADM do?**
  - GRANT Role privileges
  - CREATE, COMMENT, DROP ROLE
  - CREATE, ALTER, COMMENT, DROP TRUSTED CONTEXT
  - New DB2 10 Audit privileges
    - SELECT, INSERT, UPDATE, DELETE on new SYSIBM.SYSAUDITPOLICIES table
  - New DB2 10 row and column access controls
    - CREATE, ALTER, COMMENT, DROP row permissions and column masks
    - ALTER TABLE to activate row and column level access control
    - CREATE_SECURE_OBJECT privilege
  - SELECT, INSERT,UPDATE, DELETE on catalog tables

SECADM authority for performing security tasks

- **How to enable SECADM ?**
  - New install security zPARMs
    - **Set 2 INSTALL SECADMs** :
      - DSN6SPRM macro : SECADM1 and SECADM2
      - DSNTIPP1 install panel : SECURITY ADMIN 1 and 2 fields
    - **Set 2 SECADM_TYPE for each SECADM**
      - Values : **ROLE** or an **AUTHID.**
      - DSN6SPRM macro : SECADM1_TYPE , SECADM2_TYPE
      - DSNTIPP1 install panel : SEC ADMIN 1 TYPE and SEC ADMIN 2 TYPE fields
    - **Activated by SEPARATE_SECURITY ZParm**
      - Specified on DSNTIPP1 -
        - > **YES** – Users with SYSADM and SYSCTRL can not perform GRANTS on objects created by others
        - > **NO** – Users with SYSADM can administer security for all objects
        - > Available in CM Mode
        - > Users with INSTALL SYSADM can still perform GRANTS for other users

New authority for managing objects
without ability to access data or control access to data

- **System DBADM authority**
  - Allows the user to
    - Issue SQL CREATE, ALTER, DROP statements to manage most objects in the DB2 subsystem
      - Exception: Security objects, system objects
      - Additional privileges required to create objects such as views, functions, triggers
    - Issue most DB2 commands
    - Execute system defined stored procedures and functions
    - Access catalog tables

New authority for accessing data
without the ability to manage data or control access to data

- **DATAACCESS authority**
  - Allows the user to
    - Issue SQL SELECT, INSERT, UPDATE, DELETE statements on all user tables, views, materialized query tables
    - Execute all plans, packages and routines
    - Run RECOVERDB, REORG, REPAIR, LOAD utilities on all user databases
    - Issue ALTER and TERM UTILITY commands
    - Access catalog tables

New authority for controlling access to data
without ability to manage or access data

- **ACCESSCTRL authority**
  - Allows the user to
    - Issue SQL GRANT, REVOKE statements on most grantable privileges and administrative authorities
      - Exceptions:
      - System DBADM, DATAACCESS, ACCESSCTRL authorities
      - Security privilege, CREATE_SECURE_OBJECT
    - Access catalog tables

New authority for monitoring and tuning SQL
without ability to change or access data

- **SQLADM authority :**
  - Issue SQL EXPLAIN statement
  - Issue START, STOP and DISPLAY PROFILE commands
  - Perform actions involving
    - EXPLAIN privilege
    - STATS privilege on all user databases
    - MONITOR2 privilege
    - Execute DB2 supplied stored procedures and routines
  - Ability to SELECT, INSERT, UPDATE, DELETE on DB2 catalog tables
  - CAN NOT access data, perform DDL or EXECUTE plans or packages

NEW PRIVILEGES – EXPLAIN

- **EXPLAIN privilege**
  - Issue SQL EXPLAIN ALL statement without being able to EXECUTE that statement
  - Issue SQL PREPARE and DESCRIBE TABLE statements without having privileges on the object
  - BIND EXPLAIN(ONLY) and SQLERROR(CHECK)
  - Explain dynamic SQL statements executing under new special register
    - CURRENT EXPLAIN MODE = EXPLAIN

NEW BIND OPTIONS – EXPLAIN(ONLY) & SQLERROR(CHECK)

- **EXPLAIN(ONLY)**
  - Provides the ability to EXPLAIN statements without the ability to execute them
  - Requires EXPLAIN privilege or necessary BIND privileges
  - Populates the EXPLAIN tables without creating a package
- **SQLERROR(CHECK)**
  - Provides the ability to syntax and semantic check the SQL statements being bound without the ability to execute the statement(s)

Administrative Authority …

	Collections	User data[1]	Plans, packages & Routines	All schemas	JARS Sequences	Security
DBADM ON SYSTEM	CREATEIN	ALTER INDEX REFERENCES TRIGGER	BIND COPY	CREATEIN ALTERIN DROPIN	ALTER	
DATAACCESS		SELECT INSERT UPDATE DELETE	EXECUTE		USAGE	
ACCESSCTRL		GRANT REVOKE REVOKE BY				GRANT REVOKE REVOKE BY
SECADM	GRANT REVOKE REVOKE BY	GRANT REVOKE REVOKE BY	GRANT REVOKE REVOKE BY	GRANT REVOKE REVOKE BY	GRANT REVOKE REVOKE BY	GRANT REVOKE REVOKE BY
SQLADM		NONE				

1 – Except tables defines with row permissions or column masks

The information on this slide and the next slide can be found in the Administration Guide under the topic Administrative authorities

Administrative Authority

	Distinct types	User databases	System privileges	Catalog tables (Update when available)	Issue Commands
DBADM ON SYSTEM	USAGE privileges on BUFFERPOOLS TABLESPACE STOGROUP	CREATETAB CREATETS DISPLAYDB DROP IMAGECOPY RECOVERDB STARTDB STATS STOPDB	BINDADD BINDAGENT CREATEALIAS CREATEDBA CREATEDBC CREATETMTAB DISPLAY EXPLAIN MONITOR1/MONITOR2 SQLADM STOPALL TRACE	SELECT INSERT UPDATE DELETE	Able to issue most commands
DATAACCESS	USAGE	RECOVERDB REORG REPAIR LOAD	DEBUGSESSION	SELECT INSERT UPDATE DELETE	ALTER UTILITY
ACCESSCTRL				SELECT INSERT UPDATE DELETE	
SECADM	GRANT REVOKE REVOKE BY	GRANT REVOKE REVOKE BY		GRANT REVOKE REVOKE BY	
SQLADM		STATS	MONITOR1 MONITOR2 EXPLAIN	SELECT INSERT UPDATE DELETE	START STOP DISPLAY PROFILE

RACF support for the new Administrative Authorities

- **RACF Access Control Module ('SYS1.SDSNSAMP (DSNXRXAC)') has been enhanced to**
  - Honor the setting of SEPARATE_SECURITY
  - Implement the new DB2 administrative authorities as RACF resource checks

DB2 Authority	Resource	Class
SECADM	**<subsystem>.SECADM**	**DSNADM**
System DBADM	**<subsystem>.SYSDBADM**	**DSNADM**
DATAACCESS	**<subsystem>.DATAACCESS**	**DSNADM**
ACCESSCTRL	**<subsystem>.ACCESSCTRL**	**DSNADM**
SQLADM	**<subsystem>.SQLADM**	**MDSNSM**
EXPLAIN	**<subsystem>.EXPLAIN**	**MDSNSM**

REVOKE DEPENDENT PRIVILEGES …

- **Provides additional controls regarding cascading effects of a REVOKE statement**
  - INCLUDING DEPENDENT PRIVILEGES
  - NOT INCLUDING DEPENDENT PRIVILEGES
    - When ACCESSCTRL, DATAACCESS, or DBADM ON SYSTEM is revoked,
      - the default is always NOT INCLUDING DEPENDENT PRIVILEGES and
      - the NOT INCLUDING DEPENDENT PRIVILEGES clause must be explicitly specified

REVOKE DEPENDENT PRIVILEGES …

- ZParm – **REVOKE_DEP_PRIVILEGES**
  - Panel DSNTIPP1 –
  - Values
    - **NO, YES, SQLSTMT**
    - **NO**
      - > Dependent privileges CAN NOT be cascaded
      - You can not specify INCLUDING DEPENDENT PRIVILEGES
    - **YES** - This is pre DB2 10 behavior
      - > All revokes will include dependent privileges
      - Except when ACCESSCTRL, DATAACCESS and SYSTEMDBA are revoked
    - **SQLSTMT**
      - > Controlled at the SQL statement level as specified in the REVOKE statement. THIS IS THE DEFAULT

REVOKE DEPENDENT PRIVILEGES without cascading revokes…

```
SET CURRENT SQLID = 'DNET775';
GRANT SELECT ON DNET775.CUSTOMER TO FRANK WITH GRANT OPTION;
SELECT
  SUBSTR(GRANTOR,1,8) AS GRANTOR,
  SUBSTR(GRANTEE,1,8) AS GRANTEE,
  SELECTAUTH
FROM SYSIBM.SYSTABAUTH
WHERE TCREATOR = 'DNET775' AND TTNAME = 'CUSTOMER';
```

SYSTABAUTH

TTNAME	GRANTOR	GRANTEE	SELECTAUTH
CUSTOMER	DNET775	DNET775	G
CUSTOMER	DNET775	FRANK	G

REVOKE DEPENDENT PRIVILEGES without cascading revokes…

```
SET CURRENT SQLID = 'FRANK';
GRANT SELECT ON DNET775.CUSTOMER TO STAN;
SELECT
  SUBSTR(GRANTOR,1,8) AS GRANTOR,
  SUBSTR(GRANTEE,1,8) AS GRANTEE,
  SELECTAUTH
FROM SYSIBM.SYSTABAUTH
WHERE TCREATOR = 'DNET775' AND TTNAME = 'CUSTOMER';
```

SYSTABAUTH

TTNAME	GRANTOR	GRANTEE	SELECTAUTH
CUSTOMER	DNET775	DNET775	G
CUSTOMER	DNET775	FRANK	G
CUSTOMER	FRANK	STAN	Y

REVOKE DEPENDENT PRIVILEGES without cascading revokes and REVOKE BY

```
SET CURRENT SQLID = 'DNET775';
REVOKE SELECT ON DNET775.CUSTOMER FROM FRANK
  NOT INCLUDING DEPENDENT PRIVILEGES;
SELECT
  SUBSTR(GRANTOR,1,8) AS GRANTOR,
  SUBSTR(GRANTEE,1,8) AS GRANTEE,
  SELECTAUTH
FROM SYSIBM.SYSTABAUTH
WHERE TCREATOR = 'DNET775' AND TTNAME = 'CUSTOMER';
```

SYSTABAUTH

TTNAME	GRANTOR	GRANTEE	SELECTAUTH
CUSTOMER	DNET775	DNET775	G
CUSTOMER	FRANK	STAN	Y

```
REVOKE SELECT ON DNET775.CUSTOMER FROM STAN BY FRANK;
```

SYSTABAUTH

TTNAME	GRANTOR	GRANTEE	SELECTAUTH
CUSTOMER	DNET775	DNET775	G

DB2 Audit Capability …

- **New audit capabilities without expensive data collectors**
  - New Audit Policies are managed in the catalog
    - Audit policy provides wild carding of table names
  - Ability to audit :
    - Privileged users
    - SQL activity against a table
      - Audit policy does not require AUDIT clause to be specified
      - Audit policy generates records for all read and update access, not just first access in the transaction
      - Audit policy includes additional records identifying the specific SQL statements reading or updating an audited UTS table
    - Distributed identities

DB2 Audit Capability …

- **To create an AUDIT POLICY**
  - Insert a row into new SYSAUDITPOLICIES table
  - Specify the category and related fields
    - See SQL Reference Appendix A
  - Issue START TRACE command with audit policy name to enable audit policy
  - Issue STOP TRACE command with audit policy name to disable audit policy
  - Up to 8 audit policies can be specified to auto start when DB2 is started
  - SECADM authority required

DB2 Audit Capability …

SYSIBM.SYSAUDITPOLICIES table

Column Name	Col No	Col Type	Length
AUDITPOLICYNAME	1	VARCHAR	128
OBJECTSCHEMA	2	VARCHAR	128
OBJECTNAME	3	VARCHAR	128
OBJECTTYPE	4	CHAR	1
CREATEDTS	5	TIMESTMP	10
ALTEREDTS	6	TIMESTMP	10
CHECKING	7	CHAR	1
VALIDATE	8	CHAR	1
OBJMAINT	9	CHAR	1
EXECUTE	10	CHAR	1
CONTEXT	11	CHAR	1
SECMAINT	12	CHAR	1
SYSADMIN	13	VARCHAR	128
DBADMIN	14	VARCHAR	128
DBNAME	15	VARCHAR	24
COLLID	16	VARCHAR	128
DB2START	17	CHAR	1
IBMREQD	18	CHAR	1

DB2 Audit Capability …

- **To create a new AUDITADMIN1 policy to audit the SYSADM authority (S) and the SYSOPR authority (O), you can specify SYSADMIN as the category:**

```
INSERT INTO SYSIBM.SYSAUDITPOLICIES
   (AUDITPOLICYNAME, SYSADMIN)
 VALUES('AUDITADMIN1','OS');
```

- **You can also use the SQL LIKE predicate to audit tables of the same characteristics. For example, you can audit all tables that start with E_P in schema TSCHEMA by issuing the following INSERT statement:**
  - OBJECTTYPE ‘T’ means table
  - EXECUTE ‘C’ means Audit on all INSERT,UPDATE,DELETE statements
  - EXECUTE ‘A” means Audit all access on the table

```
INSERT INTO SYSIBM.SYSAUDITPOLICIES
   (AUDITPOLICYNAME, OBJECTSCHEMA, OBJECTNAME, OBJECTTYPE, EXECUTE)
 VALUES('TEST2','TSCHEMA','''E_P%''','T','C');
```

New improved security features provide more effective controls and accurate audit trail for remote access

- **Support distributed identities introduced in z/OS V1R11**
  - A distributed identity is a mapping between a RACF user ID and one or more distributed user identities, as they are known to application servers
- **Support client certificates and password phrases in z/OS V1R10**
  - AT-TLS secure handshake accomplishes identification and authentication when the client presents its certificate as identification and its proof-of-possession as authentication
  - A RACF password phrase is a character string made up of mixed-case letters, numbers, special characters, and is between 9 to 100 characters long
- **Support connection level security enforcement**
  - Enforces connections must use strong authentication to access DB2
  - All userids and passwords encrypted using AES, or connections accepted on a port which ensures AT-TLS policy protection or protected by an IPSec encrypted tunnel

Row and Column level access …

- **What is the purpose of row level security?**
  - Filter rows out of answer set
  - Policy can use session information, e.g. the SQL ID is in what group or user is using what role to control when row is returned in result set
  - Applicable to SELECT,INSERT, UPDATE,DELETE & MERGE
  - Application is not aware -> no need to change application
  - Defined as a row permission:

```
CREATE PERMISSION policy-name ON table-name
   FOR ROWS WHERE search-condition
   ENFORCED FOR ALL ACCESS ENABLE
```

Optimizer inserts **search condition** in all SQL statements accessing table. If row satisfies search-condition, row is returned in the answer set

Row and Column level access …

- **What is the purpose of column level security?**
  - Mask column values in answer set
  - Policy can use session information, e.g. the SQL ID is in what group or user is using what role to control when row is returned in result set
  - Defined as column masks:

```
CREATE MASK mask-name ON table-name FOR COLUMN
  column-name RETURN CASE expression ENABLE;
```

Optimizer inserts **CASE statement** in all SQL accessing table to determine mask value to return in answer set

Row and Column level access …

- **Define a column or row policy based on who is accessing the table**
  - SESSION-USER
    - Primary authorization ID of the process
  - CURRENT SQLID
    - SQL authorization ID of the process
    - SET CURRENT SQLID = some authorization id
  - VERIFY_GROUP_FOR_USER (new BIF)
    - Get authorization IDs for the value in SESSION_USER
      - Gets both primary and secondary auth ids
      - Return 1 if any of those auth IDs are in the argument

```
WHERE
  VERIFY_GROUP_FOR_USER(SESSION_USER,'MGR','PAYROLL') = 1
```

  - VERIFY_ROLE_FOR_USER (new BIF)
    - Get the role for the value in SESSION_USER
    - Return 1 if the role is in the argument list

```
WHERE
  VERIFY_ROLE_FOR_USER(SESSION_USER,'MGR','PAYROLL') = 1
```

Managing row and column access controls

- **When activated row and column access controls:**
  - Make row permissions and column masks become effective in all DML
  - All row permissions are connected with ‘OR’ to filter out rows
  - All column masks are applied to mask output
  - All access to the table if no user-defined row permissions

```
ALTER TABLE table-name
   ACTIVATE ROW LEVEL ACCESS CONTROL
   ACTIVATE COLUMN LEVEL ACCESS CONTROL;
```

- **When deactivated row and column access controls:**
  - Make row permissions and column masks become ineffective in DML
  - Opens all access to the table

```
ALTER TABLE table-name
   DEACTIVATE ROW LEVEL ACCESS CONTROL
   DEACTIVATE COLUMN LEVEL ACCESS CONTROL;
```

Row and Column level access – Banking example …

- **Only allow customer service representatives to see customer data but always with masked income**
- **Table: CUSTOMER**

ACCOUNT	NAME	PHONE	INCOME	BRANCH
1111-2222-3333-4444	Alice	111-1111	22,000	A
2222-3333-4444-5555	Bob	222-2222	71,000	B
3333-4444-5555-6666	Louis	333-3333	123,000	B
4444-5555-6666-7777	David	444-4444	172,000	C

Row and Column level access – Banking example …

- Determine access control rules for customer service rep
  - Allow access to all customers of the bank (a row permission)
  - Mask all INCOME values (a column mask)
    - Return value 0 for incomes of 25000 and below
    - Return value 1 for incomes between 25000 and 75000
    - Return value 2 for incomes between 75000 and 150000
    - Return value 3 for incomes above 150000
  - All are in the CSR group (who)
- Create a row permission for customer service representatives

```
CREATE PERMISSION CSR_ROW_ACCESS ON CUSTOMER
FOR ROWS WHERE
VERIFY_GROUP_FOR_USER (SESSION_USER,'CSR') = 1
 AND BRANCH = 'B'
ENFORCED FOR ALL ACCESS;
```

Row and Column level access – Banking example…

- **Create a column mask on INCOME for customer service rep**

```
CREATE MASK INCOME_COLUMN_MASK ON CUSTOMER
FOR COLUMN INCOME RETURN
  CASE WHEN (VERIFY_GROUP_FOR_USER (SESSION_USER,'CSR') = 1)
      THEN CASE WHEN (INCOME > 150000) THEN 3
                WHEN (INCOME > 75000) THEN 2
                WHEN (INCOME > 25000) THEN 1
           ELSE 0
      END
  ELSE 0
  END
ENABLE;
```

Row and Column level access – Banking example…

- **Activate Row-level and column-level access control**

```
ALTER TABLE CUSTOMER
  ACTIVATE ROW    ACCESS CONTROL
  ACTIVATE COLUMN ACCESS CONTROL;
```

- **What Happens in DB2?**
  - A default row permission is created implicitly to prevent all access to table customer (WHERE 1=0) except for users in the CSR group
  - All packages and cached statements that reference table CUSTOMER are invalidated

Row and Column level access – Banking example…

```
SELECT ACCOUNT, NAME, INCOME, PHONE,BRANCH
FROM CUSTOMER
WHERE BRANCH IN ('A','B');
```

ACCOUNT	NAME	INCOME	PHONE	BRANCH
2222-3333-4444-5555	Bob	1	222-2222	B
3333-4444-5555-6666	Louis	2	333-3333	B

INCOME is automatically masked by DB2

If the user is not a member of the CSR group, then no rows at all will be returned

Row and Column level access – Banking example

DB2 effectively evaluates the following revised query:

```
SELECT ACCOUNT,NAME,INCOME
   CASE WHEN(VERIFY_GROUP_FOR_USER(SESSION_USER,'CSR') = 1)
     THEN CASE WHEN(INCOME > 150000) THEN 3
               WHEN(INCOME >  75000) THEN 2
               WHEN(INCOME >  25000) THEN 1
               ELSE 0
          END
   ELSE NULL
   END AS INCOME,
   PHONE,BRANCH
FROM CUSTOMER
WHERE BRANCH IN ('A','B')
  AND ((1=0) OR (VERIFY_GROUP_FOR_USER(SESSION_USER,'CSR')) = 1
  AND BRANCH = 'B');
```

If the user is not in the GROUP CSR, the VERIFY_GROUP_FOR_USER returns 0 and no rows are returned

New Temporal Table allows DB2 to automatically maintain different versions of your data

- **Two types of time sequences of table rows are supported through the introduction of database defined time periods**
  - **SYSTEM_TIME** is used for system maintained history for a new concept of “versioning” which archives old rows into a history table
  - **BUSINESS_TIME** is a period that represents when a row is valid to the user or application. The BUSINESS_TIME period can be used to model data in the past, present, and future as the data values are controlled by the application
  - A **bitemporal** table can includes both types of periods

Defining system versioning on a table

- **System versioning is implemented by altering an existing or creating a table with two timestamps, a history table, and defining the versioning relationship between tables**
- **System period temporal table must have:**
  - SYSTEM_TIME is defined as two TIMESTAMP(12) NOT NULL columns
  - First column defines the row begin time
  - Second column defines the row end time
  - Third column defined as TIMESTAMP(12) for the transaction that created row
    - example of period definition: PERIOD SYSTEM_TIME(col1, col2)
- **History table must have:**

  - Same number of columns as the system period temporal table
  - All columns must have the same corresponding names, data type, null attribute, ccsid, subtype, hidden attribute and fieldproc as the system period temporal table

Add system versioning to a table to audit changes

- **After the two tables are appropriately defined:**
  - ALTER TABLE table-name ADD VERSIONING is specified on the base table that is to be versioned, not the history table
- **To query historical data, the table-reference of the FROM clause is extended to request historical data**
  - DB2 rewrites the user's query to include data from the history table with a UNION ALL operator
- **New FROM SYSTEM_TIME clauses:**
  - table-name FOR SYSTEM_TIME **AS OF** timestamp-expression
  - table-name FOR SYSTEM_TIME **FROM** timestamp-expression1 **TO** timestamp-expression2
  - table-name FOR SYSTEM_TIME **BETWEEN** timestamp-expression1 **AND** timestamp-expression2

SYSTEM_TIME Period Example

Base Table

```
CREATE TABLE     policy_info
(policy_id  CHAR(4)                 NOT NULL,
coverage    INT                     NOT NULL,
sys_start   TIMESTAMP(12)           NOT NULL   GENERATED ALWAYS AS ROW BEGIN,
sys_end     TIMESTAMP(12)           NOT NULL   GENERATED ALWAYS AS ROW END,
create_id   TIMESTAMP(12)                      GENERATED ALWAYS AS TRANSACTION START ID,
PERIOD      SYSTEM_TIME(sys_start, sys_end));
```

SYSTEM TIME columns

History Table

```
CREATE TABLE     hist_policy_info
(policy_id  CHAR(4)                 NOT NULL,
coverage    INT                     NOT NULL,
sys_start   TIMESTAMP(12)           NOT NULL,
sys_end     TIMESTAMP (12)          NOT NULL,
create_id   TIMESTAMP(12));
```

To enable SYSTEM TIME you then alter the table:

```
ALTER TABLE policy_info
ADD VERSIONING USE HISTORY TABLE hist_policy_info;
```

INSERT a row into a SYSTEM_TIME table

At timestamp '2009-01-05-12:12:12.001122000000',

INSERT INTO policy_info (policy_id, coverage)
VALUES('A123', 12000);

Policy_info

Policy ID	Coverage	Sys_start	Sys_end
A123	12000	2009-01-05-12.12.12.001122000000	9999-12-31-24.00.00.000000000000

Hist_policy_info

Policy ID	Coverage	Sys_start	Sys_end

UPDATE a row in a SYSTEM_TIME table

At timestamp '2009-01-09-12:12:12.012345000000',

```
UPDATE policy_info SET coverage = 15000
        WHERE policy_id = 'A123';
```

Policy_info

Policy ID	Coverage	Sys_start	Sys_end
A123	15000	2009-01-09-12.12.12.0123450000000	9999-12-31-24.00.00.000000000000

Hist_policy_info

Policy ID	Coverage	Sys_start	Sys_end
A123	12000	2009-01-05-12.12.12.001122000000	2009-01-09-12.12.12.012345000000

Query a row in a SYSTEM_TIME table

Policy_info

Policy ID	Coverage	Sys_start	Sys_end
A123	15000	2009-01-09-12.12.12.012345000000	9999-12-31-24.00.00.000000000000

Hist_policy_info

Policy ID	Coverage	Sys_start	Sys_end
A123	12000	2009-01-05-12.12.12.001122	2009-01-09-12.12.12.012345000000

```
SELECT policy_id, coverage FROM policy_info
        FOR SYSTEM_TIME
        AS OF '2009-01-08-00:00:00.000000000000';
```

- Query returns the row of ('A123', 12000)

DB2 10 for z/OS Security Enhancements

Help Satisfy Your Auditors using new features

- New granular authorities to reduce data exposure for administrators
- New auditing features using new audit policies comply with new laws
- New row and column access table controls to safe guard your data
- New temporal data to comply with regulations to maintain historical data

Questions?