Designing a Modern Database

on IBM i

Paul Tuohy
ComCon
System *i* Developer
5, Oakton Court,
Ballybrack
Co. Dublin
Ireland

Phone: +353 1 282 6230
e-Mail: paul@systemideveloper.com
Web: www.systemideveloper.com
www.ComConAdvisor.com

IBM CHAMPION

System *i* Developer
The Summit of Education Excellence

ComCon

Agenda

System i Developer
The Summit of Education Excellence
ComCon

- Data Modeling
- Identity Columns
- Constraints
- Views
- Triggers
- Other Features
  - Auto Generated Values
  - Lookup Tables

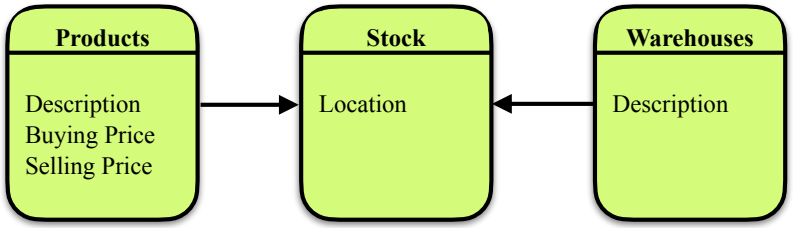
Data Modeling

Data Modeling

The process of developing data model for data to be stored in a Database

Ensure consistency in

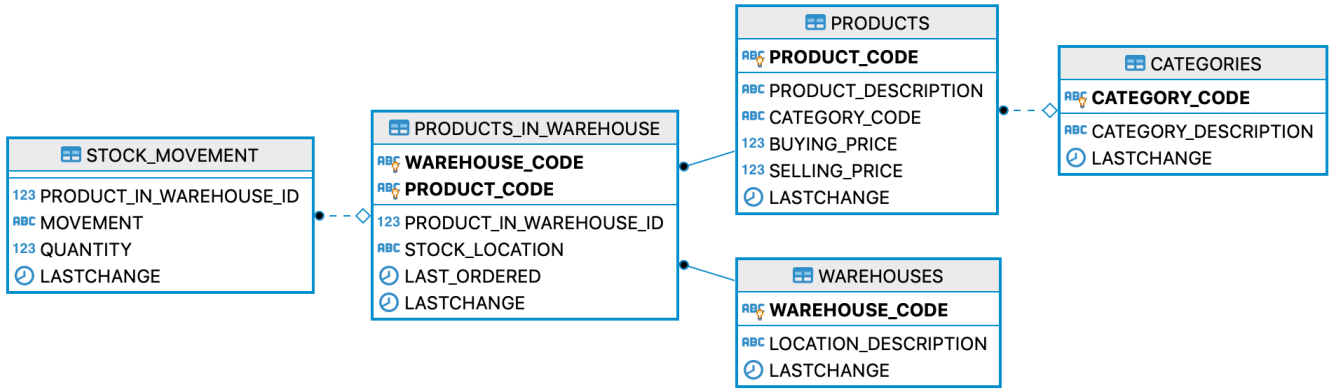
- Naming conventions
- Default values
- Semantics
- Security



Ensures quality of the data

Helps to define

- Relational tables
- Primary and foreign keys



Data Models

- Conceptual
  - An organized view of database concepts and their relationships
- Logical
  - Define the structure of data elements and set relationships between them
- Physical
  - Describes the database specific implementation of the data model

Conceptual Data Model

Represent data as a user will see it in the "real world"

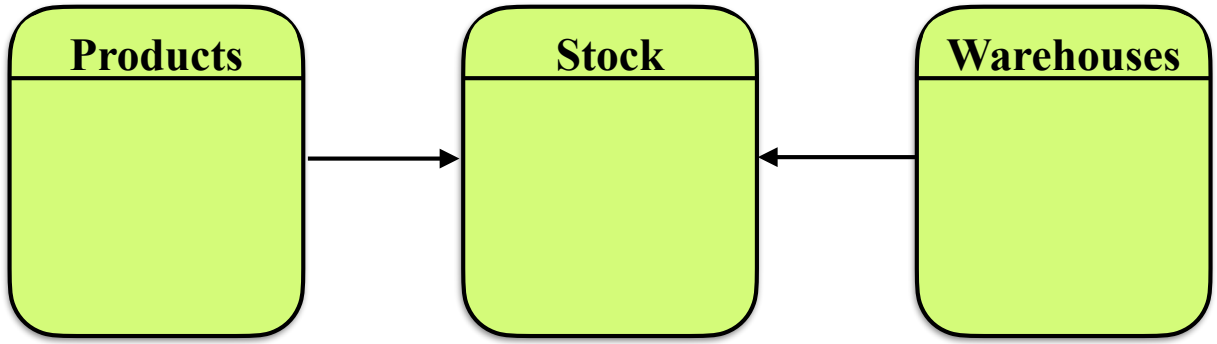
Developed independently of hardware, storage and software

Designed and developed for a business audience

- Organisation-wide coverage of the business concepts

Establishes

- Entities: A real-world thing
- Attributes: Characteristics or properties of an entity
- Relationships: Dependency or association between entities



Logical Data Model

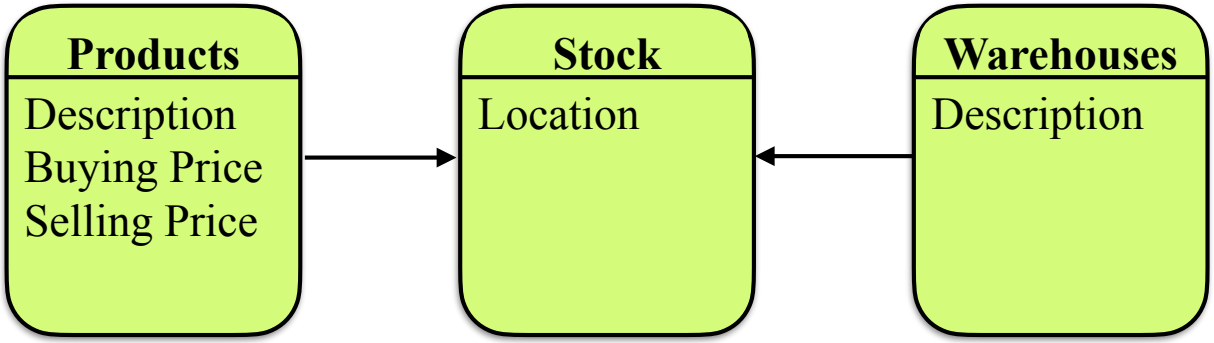
Describes data needs for a project

Designed and developed independently from the DBMS

Data attributes will have datatypes with exact precisions and length.

Normalization processes

- Typically till 3rd Normal Form



Physical Data Model

Relationships between tables

- Addresses cardinality and nullability of the relationships

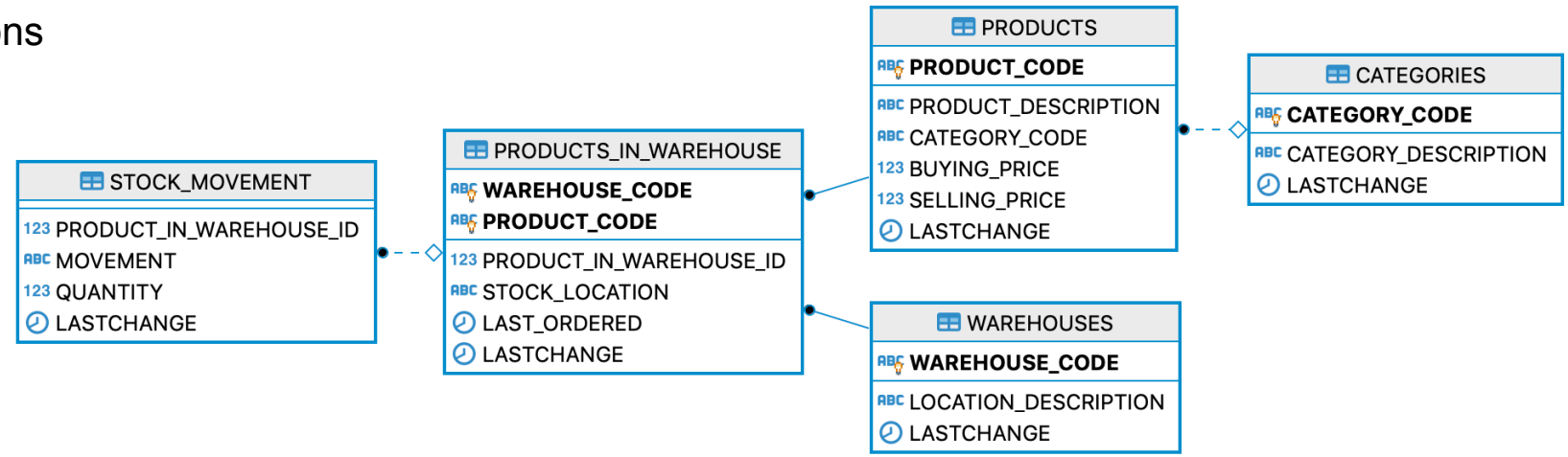
Developed for a specific version of a DBMS

- Location, data storage or technology to be used in the project

Columns have exact datatypes, lengths and default values assigned

Definition of

- Primary and Foreign keys
- Views
- Indexes
- Authorizations



What a Data Model Helps You Avoid

Relationship rules implemented in programs

Business data is

- Missing
- Scattered across different tables
- Not in a very usable format

Redundant data across multiple tables

Data types and attributes for related columns don't match

Relationships defined by multiple columns

Sensitive values being used as keys

Normalization (to 3NF)

System i Developer
The Summit of Education Excellence
ComCon

First Normal Form (1NF)

- No duplicate rows
  - Define a unique key
- No duplicate columns
  - No arrays
- Separate tables for each group of related data

Second Normal Form (2NF)

- Columns relate to the complete key
- Define table relationships with foreign keys

Third Normal Form (3NF)

- Usually good enough
- Remove columns that are not directly dependent upon the primary key

Basically, normalization ensures that all data is dependent on

- The Key
- The whole Key
- And nothing but the Key

Tools

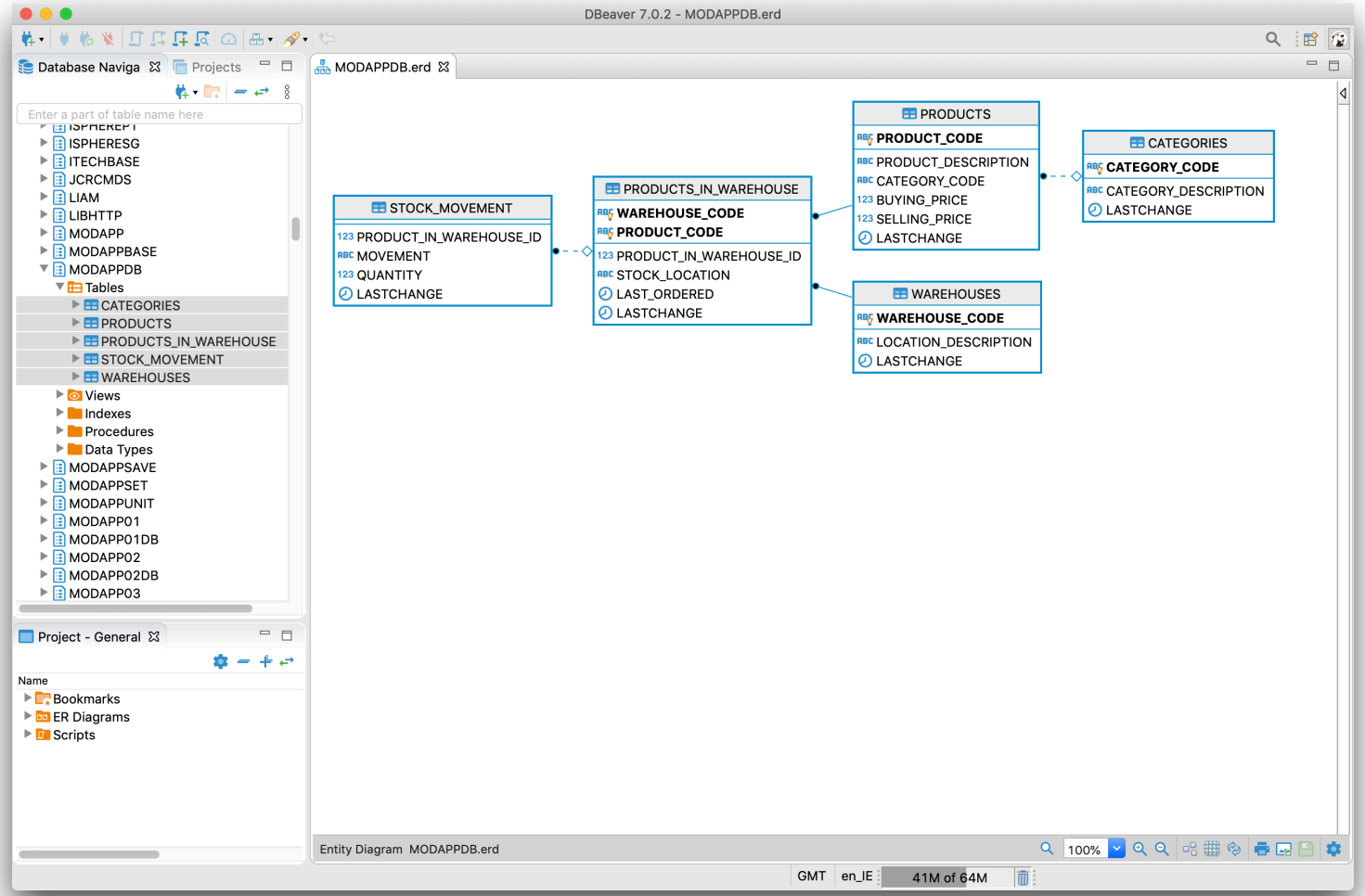
Lots of Choices

Data Studio

- Eclipse Based
- Windows only

DBeaver

And lots of others



Identity Columns

Identity Columns

System i Developer
The Summit of Education Excellence
ComCon

A column with an auto-generated numeric value

- New value assigned when row is inserted
- Requires a constraint to ensure uniqueness

A generated key

Can be used as generated key value

An associative key

- Actual key is multiple columns
- Associative tables contain identity column value
  - As opposed to duplicating the multiple key columns
- Joins to associative tables are on identity column
  - As opposed to the multiple key columns

Can be used as alternative to “sensitive” key column

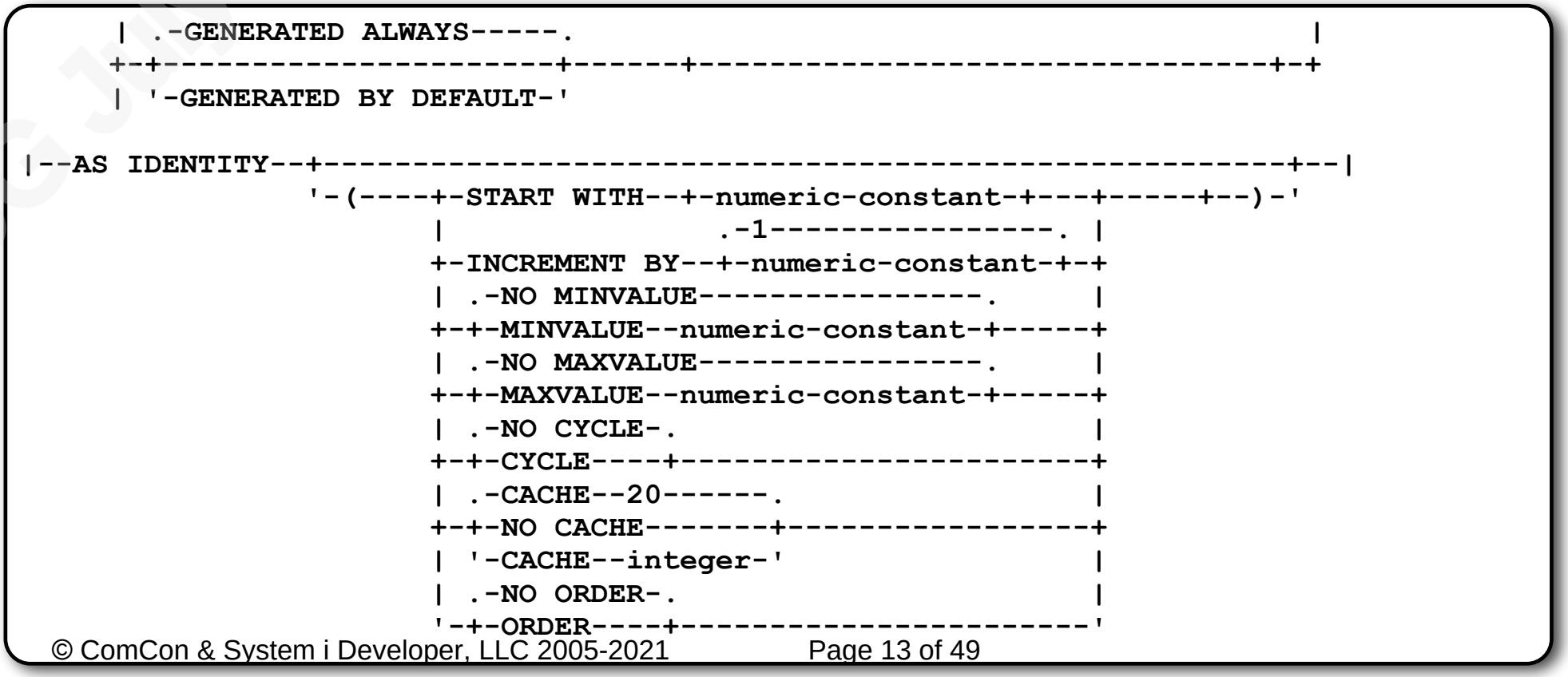
- e.g. An alternative to a social security number

Identity Columns

System i Developer
The Summit of Education Excellence
ComCon

Specified as part of the column definition on the table

```
create table EMPLOYEES
 (employee_id for column empid integer GENERATED ALWAYS AS IDENTITY,
  name varchar(30),
  department_no for column deptno char(3),
  constraint PK_EMPLOYEES primary key( employee_id ));
```



Identity Columns

Automatic generation of a unique, numeric value for every row in a table

EMPLOYEE_ID	NAME	DEPARTMENT_NO
1	Paul	A00
2	Susan	A00
3	Jon	B01

```
insert into employees
   values (default, 'Paul', 'A00'),
          (default, 'Susan', 'A00');

insert into employees (name, department_no)
   values ('Jon', 'B01');

select * from employees;
```

Identity Columns and RPG

Identity is ALWAYS generated for non-SQL interface

Use IDENTITY_VAL_LOCAL() to determine generated identity

EMPLOYEE_ID	NAME	DEPARTMENT_NO
1	Paul	A00
2	Susan	A00
3	Jon	B01
4	Amy	C01
5	Terry	C01

```
dcl-f fid_emp_i1 disk usage(*input: *update: *delete: *output) keyed;

dcl-s lastkey int(10);

name = 'Amy';
deptno = 'C01';
write id_emp;
exec sql
values identity_val_local() into :lastkey;

empid = 3;                  // Not the value put in the row!!!!!
name = 'Terry';
deptno = 'C01';
write id_emp;
exec sql
values identity_val_local() into :lastkey;
*inLR = *on;
```

Associative Key

Identity column used for “complex” key

Primary and unique keys defined with table

```
CREATE OR REPLACE TABLE PRODUCTS_IN_WAREHOUSE FOR SYSTEM NAME PRODWARE (
  WAREHOUSE_CODE          FOR COLUMN WARECODE   CHAR(2) NOT NULL DEFAULT '' ,
  PRODUCT_CODE            FOR COLUMN PRODCODE   CHAR(7) NOT NULL DEFAULT '' ,
  PRODUCT_IN_WAREHOUSE_ID FOR COLUMN PRODWAREID INTEGER GENERATED ALWAYS AS IDENTITY
  STOCK_LOCATION          FOR COLUMN STOCKLOC  CHAR(5) NOT NULL DEFAULT '' ,
  LAST_ORDERED            FOR COLUMN LASTORDER TIMESTAMP(6)
                          NOT NULL DEFAULT CURRENT TIMESTAMP ,
  LASTCHANGE TIMESTAMP(6) GENERATED ALWAYS FOR EACH ROW ON UPDATE
                            AS ROW CHANGE TIMESTAMP NOT NULL NOT HIDDEN,
  CONSTRAINT PK_PRODUCTS_IN_WAREHOUSE_ALL0008
                          PRIMARY KEY( WAREHOUSE_CODE, PRODUCT_CODE ),
  CONSTRAINT UK_PRODUCTS_IN_WAREHOUSE_ALL0009 UNIQUE ( PRODUCT_IN_WAREHOUSE_ID ) )
  RCDFMT PRODWARER  ;
```

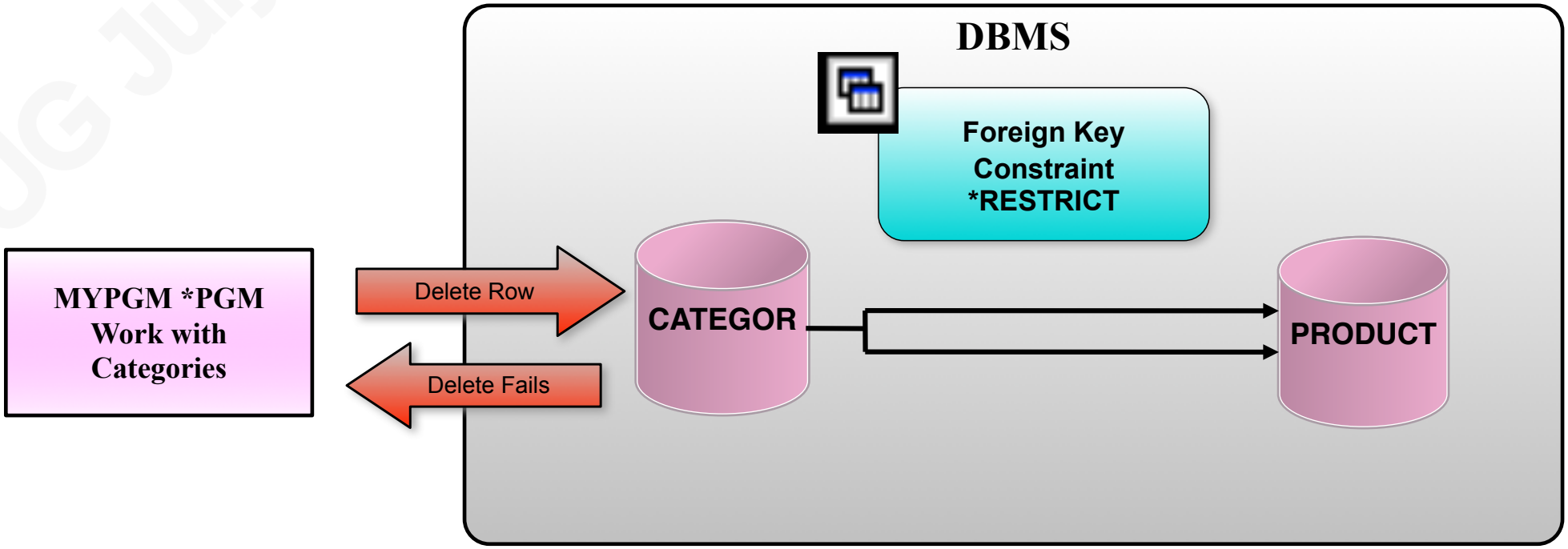
```
CREATE OR REPLACE TABLE STOCK_MOVEMENT FOR SYSTEM NAME STOCKMOVE (
  PRODUCT_IN_WAREHOUSE_ID FOR COLUMN PRODWAREID INTEGER,
  MOVEMENT   CHAR(2) NOT NULL DEFAULT '' ,
  QUANTITY   DECIMAL(7, 0) NOT NULL DEFAULT 0 ,
  LASTCHANGE TIMESTAMP(6) GENERATED ALWAYS FOR EACH ROW ON UPDATE
                            AS ROW CHANGE TIMESTAMP NOT NULL NOT HIDDEN)
  RCDFMT STOCKMOVER ;
```

Constraints

Constraints

The ability of the database management system to ensure:

- logical consistency of data values between tables
- validity of data relationships
- validity of column contents
- robust enforcement of integrity constraints



Key Constraints

Identify key requirements for a table

- One primary key
- Multiple unique keys

A key constraint results in an access path

- In the same way as an index or a logical file
- But an object is not created for a key constraint
- Results in an entry in the System Catalog

Parent must have a key constraint to be used in a Foreign Key Constraint

- More later

Defining Key Constraints in SQL

Key constraints may be defined

- As a clause of CREATE TABLE
- Using ALTER TABLE


```
CREATE TABLE CATEGOR (
          CATCOD CHAR(2) CCSID 37 NOT NULL DEFAULT '' ,
          CATDES CHAR(20) CCSID 37 NOT NULL DEFAULT '' ,
          CONSTRAINT PK_CATEGORIES PRIMARY KEY( CATCOD ) )
          RCDFMT CATEGORR    ;

LABEL ON CONSTRAINT PK_CATEGORIES IS 'Primary Key for Categories' ;
```

```
ALTER TABLE CATEGOR ADD CONSTRAINT PK_CATEGORIES PRIMARY ( CATCOD ) ;
```

Foreign Key Constraints

System i Developer
The Summit of Education Excellence
ComCon

Defines a relationship between a Dependent and Parent table

- What happens to Dependent Rows when a Parent row is altered
- Define rules for Update and Delete
- Insert rule is implicit

Basic requirement:

- Parent table requires a primary or unique key constraint for the relationship
- Parent key and foreign key must have matching field attributes

Parent and Dependent tables must be

- Single member
- Externally described

Journal requirements

- Restrict rule: journal not required
- Any other rule:
  - Parent and Dependent tables must be journaled to the same journal
  - Implicit commitment control is started automatically

Defining Foreign Key Constraints in SQL

Foreign key constraints may be defined

- Using ALTER TABLE


```
ALTER TABLE PRODUCT
          ADD CONSTRAINT FK_PRODUCTS_TO_CATEGORIES
          FOREIGN KEY( CATCOD )
          REFERENCES MODERNIZE.CATEGOR ( CATCOD )
          ON DELETE RESTRICT
          ON UPDATE RESTRICT ;
```

Foreign Key Constraint Rules

Delete rule

- RESTRICT - Row cannot be deleted if there are dependent rows
- CASCADE - OK to delete a parent row, but all dependent rows are deleted as well
- SET NULL - Null-capable columns, in the dependent key, are set to null
- SET DEFAULT - Columns in the dependent key are set to their default values
- NO ACTION - Row cannot be deleted if there are dependent rows
  - However, triggers will be fired BEFORE checking referential constraints

Update rule

- Parent key values cannot be changed if dependent rows exist
- RESTRICT - Check for dependent rows immediately
- NO ACTION - Fire triggers before checking for dependent rows

Insert rule

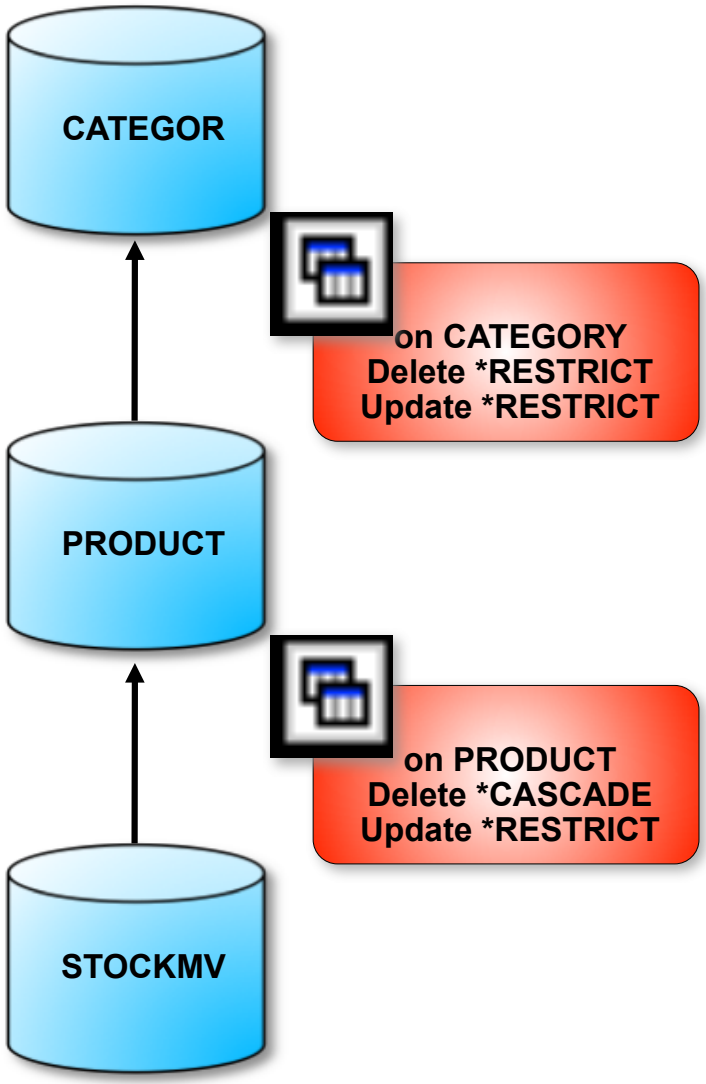
- Rule is implicit
- Rows cannot be inserted into dependent without a matching parent row

Journaling of parent and dependent tables is required

- For any rule in effect EXCEPT RESTRICT
- Must be attached to the same journal
- Allows the DBMS to roll back changes if transaction cannot be completed

A Referential Integrity Network

Add a Foreign Key Constraint between the product and stock movement tables



Access Paths and Constraints

DBMS requires access paths for key and foreign key constraint enforcement

At creation time, the system will

- Share an existing access path with matching attributes
- Create a new access path, if sharing is not possible

Access Paths created/used

- Primary key
  - Unique key access path
  - Only one is possible per physical file
- Unique key
  - Unique key access path
  - Allows same table to be a parent in several relationships
    - With different key requirements
- Foreign Key Constraint
  - Non-unique key access path based on a dependent table

A Constraint access path is

- Part of the object
  - Saved/restored with the object
- Considered by the Query Optimizer

Check Constraints

Check Constraints allow you to specify limits for columns

- Effectively an SQL WHERE Clause
- Constraint is checked when row is inserted or updated
- Nearest equivalent is COMP, RANGE and VALUES in DDS for DSPF
  - But MUCH more powerful

Some Check Constraints for the Product Table

- Description may not be blank
- The Selling Price must be greater than the Buying Price

Defining Check Constraints in SQL

Check constraints may be defined

- Using ALTER TABLE


```
ALTER TABLE PRODUCT
          ADD CONSTRAINT CHECK_PRODUCT_DESCRIPTION_ERR0001
          CHECK( PRODDS <> ' '  ) ;

LABEL ON CONSTRAINT CHECK_PRODUCT_DESCRIPTION_ERR0001
   IS 'Product Description Must be Entered' ;

ALTER TABLE PRODUCT
          ADD CONSTRAINT CHECK_BUYING_EXCEEDS_SELLING_ERR0002
          CHECK( SELLPR > LNDCST  ) ;

LABEL ON CONSTRAINT CHECK_BUYING_EXCEEDS_SELLING_ERR0002
   IS 'Buying Price Must Exceed Selling Price' ;
```

Managing Constraints

Constraints may not be changed

- Must be removed and new constraint added

Constraints may be disabled/enabled

- Useful in a batch environment
  - Helps speed process
  - Constraint not checked on every row action
  - Constraint checked for all rows when re-enabled
- May specify that data is not checked when a constraint is enabled

A Check Pending Status can come about in a number of ways:

- Define a constraint for a table that already contains data
  - And constraint violations exist in the data
- Temporarily disable constraints for a batch process
  - For performance reasons
  - Everything didn't go as expected!
- Tables are restored
  - Some rows do not comply with the constraint
- Apply Journal Changes (APYJRNCHG) or Remove Journal Changes (RMVJRNCHG)
  - Again resulting in rows that do not comply with the constraint

Tables in a check pending status must be corrected immediately

Monitoring Exceptions in Applications

Applications should handle new error messages/status codes

- Received on a WRITE, UPDATE or DELETE operation
- System messages:
  Notify: CPF502D, CPF502E, CPF502F, CPF503A, CPF523B
  Escape: CPF523B, CPF523C

RPG programs

- Use E extender/%Error() BIF and check for %Status() codes 01222 and 01022

Other Languages

- ILE COBOL - File status 9R
- ILE C - Mapped to existing error numbers
- OPM programs - Mapped to existing error codes

Use the QMHRCVPM API to retrieve the constraint name in the message data

- Use format RCVM0100

SQL

- SQLCODES 530, 531, 532
- Constraint name in SQLERRMC (in SQLCA)

Trapping a Constraint in RPG

This is for Native I/O only

Status code simply indicates a constraint violation

- It doesn't tell you which constraint
- That requires a little more work
  - sendConstraintMessage() below
  - Full subprocedure in notes

```
dcl-C IS_DUPLICATE    01021;
dcl-C IS_CONSTRAINT_1 01022;
dcl-C IS_CONSTRAINT_2 01222;
dcl-C IS_TRIGGER_1    01023;
dcl-C IS_TRIGGER_2    01024;

if (status = IS_DUPLICATE);                  // Duplicate
   uadd_Message(ERR_DUPLICATE);
elseIf (status = IS_CONSTRAINT_1)  Or        // Referential Constraint
       (status = IS_CONSTRAINT_2);
   ucheck_ConstraintMessage();
elseIf (status = IS_TRIGGER_1)  Or           // Trigger
       (status = IS_TRIGGER_2);
   uadd_Message(ERR_TRIGGER);
else;                                        // Other
   uadd_Message(ERR_UNKNOWN);
   return *On;
endIf;
```

SendConstraintMessage() Subprocedure

Constraint name is in the second level message text

- Use the QMHRCVPM API to read back through the program message queue and extract the constraint name

A technique for meaningful messages

- Place message it as last seven characters of constraint name
  - CHECK_PRODUCT_DESCRIPTION_ERR0001
  - CHECK_BUYING_EXCEEDS_SELLING_ERR0002
  - FK_PRODUCTS_TO_CATEGORIES_ERR0003

```
receiveMsg( msgBack: %size(msgBack): 'RCVM0100': '*': 2: '*PRV'
           : setMsgKey: 0: '*SAME': APIError);
If (msgBack.msgId = 'CPF502D' Or msgBack.msgId = 'CPF502E' Or
    msgBack.msgId = 'CPF502F' Or msgBack.msgId = 'CPF503A' Or
    msgBack.msgId = 'CPF503B');
    constraint = %subst(msgBack.msgData:177);
    monitor;
       msgId = %subst(constraint:%scan(' ':constraint)-7);
       addMessage(MsgId);
       return;
    on-Error;
       addMessage(ERR_CONSTRAINT);
       return;
    endMon;
endIf;
```

Views

It Is All About Views

System i Developer
The Summit of Education Excellence
ComCon

What is a View?

- A “permanent” SELECT statement
  - Without an ORDER BY clause
- Exists as a *FILE object
- Can be treated as a table

Access to Data should be through views

- Not directly to the table

Minimize repeated code

- Joins
- Calculated columns
  - Casting columns
  - Use of scalar functions
- Grouping

Security

We can have a view of a view

Reduces number of items in FROM clause

There is NO overhead to using views

No maintenance or run-time overhead

Classifying Views

Simple (or Primary) View

- Over a single table
- Used for Insert, Update, Delete

List Views

- List joined data
  - e.g. Order Header joined to Customer Master to list Customer name

Structural Views

- Modularize, Grouping , Standard joins etc.

Reporting Views

- Web Query
- For export to Data Warehouse

Program Views

- Reduce complexity of SQL embedded in program

Consider a naming convention

- SV_, LV_, TV_, RV_, PV_

Triggers

Triggers

A Trigger is a program that is called whenever a specified event occurs on a table

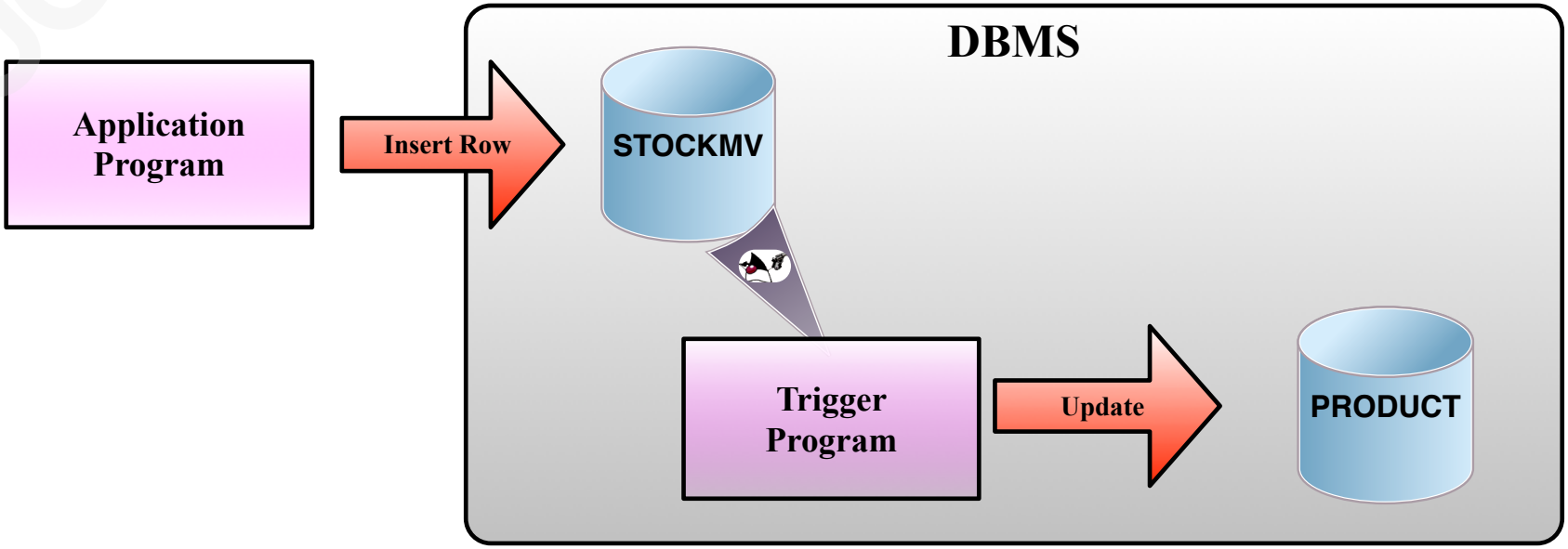
- The Trigger Program is called by the DBMS

When might you use triggers?

- To consistently enforce complex business rules
- To monitor critical data
- Validate data or security beyond object level
- Spawn processes in a client-server environment

Triggers may be

- External Triggers or SQL Triggers



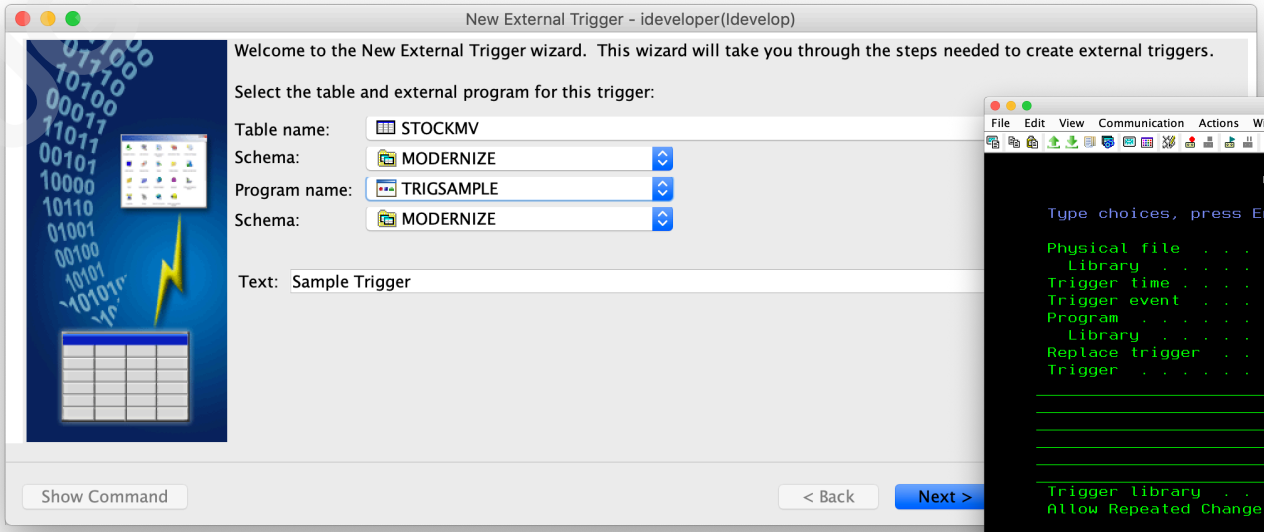
How Are Triggers Managed?

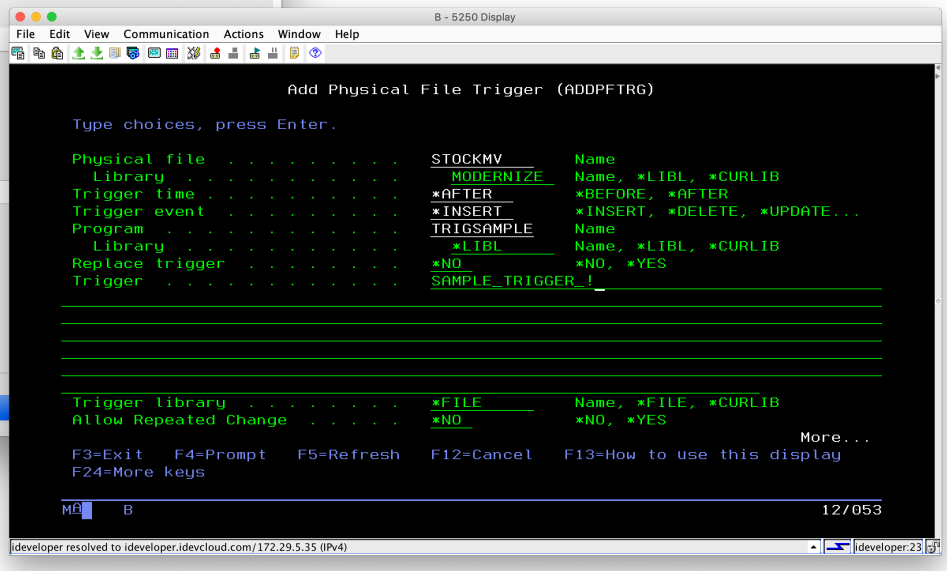
In ACS

- From context menu of table, select
  - *New->Trigger->External* or *New->Trigger->SQL*
- In schema sub menu, from context menu of Triggers select
  - *New->External* or *New->SQL*

Using CL commands

- Add Physical File Trigger (ADDPFTRG)
- Change Physical File Trigger (CHGPFTRG)
- Remove Physical File Trigger (RMVPFTRG)





Components

Triggers are user-written programs

- Associated with a table or physical file
- Activated by the DBMS before or after a database change
- Independent from applications
- Can be developed with any system compiler or SQL

The base table

- The table (or physical file) to which the trigger is attached

The trigger event

- What causes the trigger to be called
- Row events of insert, delete, update or read
  - However, Read triggers can have severe performance impact - use with Caution
- Column events available with SQL Triggers only

The trigger time

- Timing in relation to the trigger event
  - Before or after the trigger event

The Base Table

Must be a table or a physical file

- Triggers may not be attached to indexes, views or logicals
- However, changes initiated through indexes, views or logicals will cause a trigger, attached to the based table/physical file, to fire

When saved, trigger information is saved as well

- Trigger information is stored with the file object
  - e.g., Trigger program name, trigger event, trigger time
- Trigger programs, however, are NOT saved with the file object

If trigger program is renamed/moved/deleted, file description information is not updated

- You must remove and add the trigger again

The Trigger Event and Time

System i Developer
The Summit of Education Excellence
ComCon

An Event

- A row is read, inserted, updated or deleted
- A column is changed (SQL triggers only)

Time

- Before or after the Event

Triggers are attached based on a combination of Events and Times

- e.g. Before Insert, After Update etc.

The same Trigger program may be attached to multiple tables

A trigger program can be specified for any combination of Trigger event and time

- Up to 300 triggers per table

The Trigger Event and Time

Table level events are not supported

Before triggers are capable of stopping the event

Before triggers can also change/update the row image

- If the "Allow Repeated Change - *YES" option is specified

After triggers may not stop the event

- Unless using commitment control

External Trigger Program

An application program (*PGM object)

- Written in any language

May do almost anything any other program can do

Is passed a standard parameter block

- First parameter contains information about the row and the changes
  - Typically coded in RPG as a Data Structure
  - Often called the "Trigger Buffer"
  - Contains a fixed length part and a variable length part
    - Contains the before and/or after images of the row
- Second parameter contains the length of first parameter

External Triggers may only be specified for row events

SQL Triggers may be specified for row or column events

Other Bits and Pieces

Nulls and VARCHAR

System i Developer
The Summit of Education Excellence
ComCon

Columns defined as Not Null

- Only use Null if truly necessary

Description is VARCHAR

- Removes requirement for trimming


```
CREATE OR REPLACE TABLE PRODUCTS (
  PRODUCT_CODE         FOR COLUMN PRODCODE  CHAR(7) NOT NULL DEFAULT '' ,
  PRODUCT_DESCRIPTION FOR COLUMN PRODDESC VARCHAR(50) ALLOCATE(50)
                       NOT NULL DEFAULT '' ,
  CATEGORY_CODE        FOR COLUMN CATCODE   CHAR(2) NOT NULL DEFAULT '' ,
  BUYING_PRICE         FOR COLUMN BUYPRICE  DECIMAL(9, 2) NOT NULL DEFAULT 0 ,
  SELLING_PRICE        FOR COLUMN SELLPRICE DECIMAL(9, 2) NOT NULL DEFAULT 0 ,
  LASTCHANGE TIMESTAMP(6) GENERATED ALWAYS FOR EACH ROW ON UPDATE
                       AS ROW CHANGE TIMESTAMP NOT NULL NOT HIDDEN,
  CONSTRAINT PK_PRODUCTS_ALL0006 PRIMARY KEY( PRODUCT_CODE ) )
  RCDFMT PRODUCTSR  ;
```

Autogenerated Column Values

Auto-Generated Values

- Identity Columns
- Row Change Timestamps
- Data Change Operation
- Special Registers / Built-in Global Variables

May effect performance

- Effectively disables blocking
- Use with care
- Evaluate the effect

```
CREATE OR REPLACE TABLE PRODUCTS (
  PRODUCT_CODE          FOR COLUMN PRODCODE  CHAR(7) NOT NULL DEFAULT '' ,
// More columns here
  LASTCHANGE TIMESTAMP(6)
             GENERATED ALWAYS FOR EACH ROW ON UPDATE
             AS ROW CHANGE TIMESTAMP NOT NULL NOT HIDDEN,
  RECORD_TYPE_CHANGE   FOR COLUMN REC_CHANGE CHAR (1)
             GENERATED ALWAYS AS (DATA CHANGE OPERATION)
  RECORD_USER          FOR COLUMN REC_USER VARCHAR(100)
             GENERATED ALWAYS AS (SESSION_USER)
  RECORD_CLIENT_IP     FOR COLUMN REC_IP VARCHAR(20)
             GENERATED ALWAYS AS (SYSIBM.CLIENT_IPADDR)
  CONSTRAINT PK_PRODUCTS_ALL0006 PRIMARY KEY( PRODUCT_CODE ) )
  RCDFMT PRODUCTSR  ;
```

Look Up Tables

An auxiliary table that holds static data

- Country Codes
- State Codes

Can be faster to join to a Look Up table as opposed to calculating a value

- Date information
  - ISO Format
  - Locale Format
  - Julian Day
  - Public Holiday
  - Month End
  - Etc.

Wrap Up

Make use of a data model

Normalize your database

Make use of Identity Columns

Make use of Constraints

Make use of Triggers

Master Views

LET THE DATABASE DO THE WORK FOR YOU

iTalk with Tuohy

Check it out at ibmsystemsmag.com/ibmi/trends/iTALK-WITH-TUOHY/