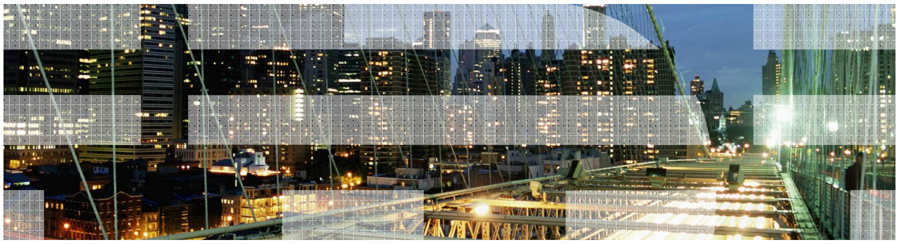
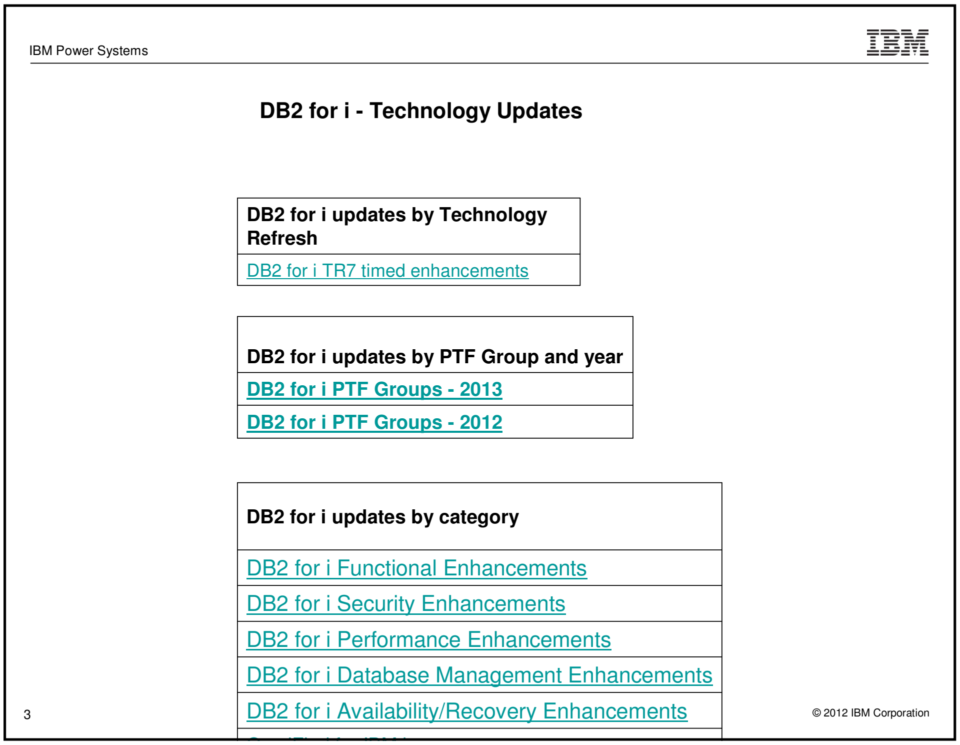
DDS vs DDL - Why Modernize with SQL

Linda M Swan
lmswan@us.ibm.com
IBM Rochester MN Lab



“New” DB2 for i Web Resources

- Regularly check (or Subscribe) to the DB2 for i Technology Updates Wiki!
  - Contains details on new PTFs that deliver new DB2 capabilities
  - Examples:
    - PROGRAM NAME keyword for controlling SQL Triggers Program Name
    - SQL Query Engine 6.1 support for Logical File on FROM clause
    - CONNECT BY 7.1 support for hierarchical queries
    - RUNSQL CL command
    - Multiple event Triggers
  - Wiki URL:
    https://www.ibm.com/developerworks/ibmi/techupdates/db2
- The wiki is part of a IBM i zone in IBM developerWorks
  https://www.ibm.com/developerworks/ibmi/
- DB2 for i Blog: http://db2fori.blogspot.com/

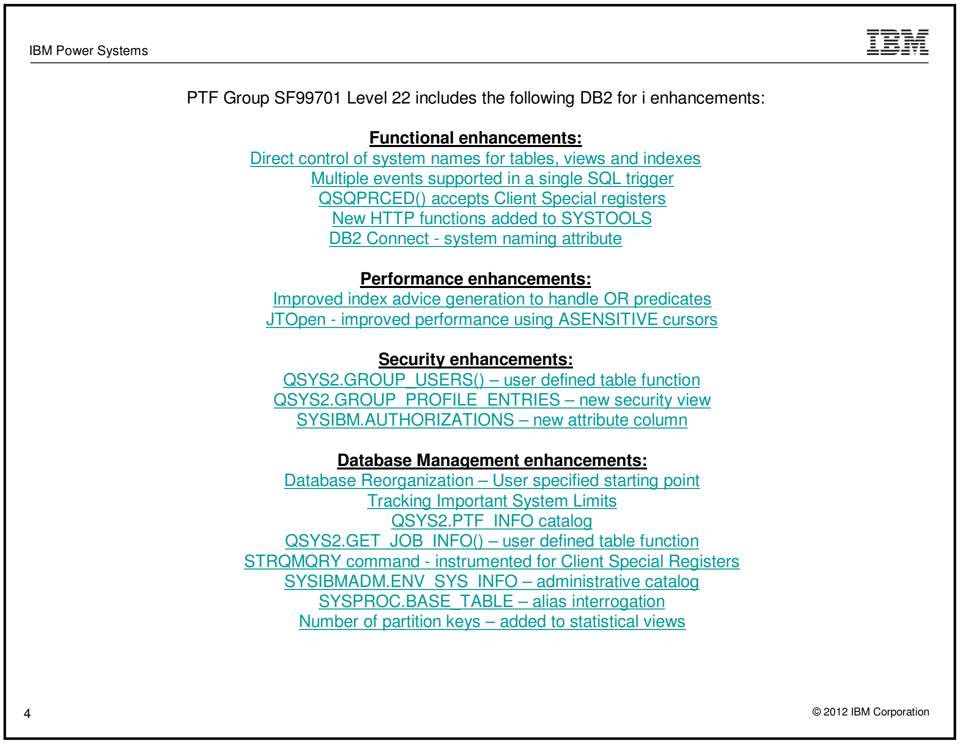


DB2 for i - Technology Updates

DB2 for i updates by Technology Refresh
DB2 for i TR7 timed enhancements

DB2 for i updates by PTF Group and year
DB2 for i PTF Groups - 2013
DB2 for i PTF Groups - 2012

DB2 for i updates by category
DB2 for i Functional Enhancements
DB2 for i Security Enhancements
DB2 for i Performance Enhancements
DB2 for i Database Management Enhancements
DB2 for i Availability/Recovery Enhancements



PTF Group SF99701 Level 22 includes the following DB2 for i enhancements:

Functional enhancements:
Direct control of system names for tables, views and indexes
Multiple events supported in a single SQL trigger
QSQPRCED() accepts Client Special registers
New HTTP functions added to SYSTOOLS
DB2 Connect - system naming attribute

Performance enhancements:
Improved index advice generation to handle OR predicates
JTOpen - improved performance using ASENSITIVE cursors

Security enhancements:
QSYS2.GROUP_USERS() – user defined table function
QSYS2.GROUP_PROFILE_ENTRIES – new security view
SYSIBM.AUTHORIZATIONS – new attribute column

Database Management enhancements:
Database Reorganization – User specified starting point
Tracking Important System Limits
QSYS2.PTF_INFO catalog
QSYS2.GET_JOB_INFO() – user defined table function
STRQMQRY command - instrumented for Client Special Registers
SYSIBMADM.ENV_SYS_INFO – administrative catalog
SYSPROC.BASE_TABLE – alias interrogation
Number of partition keys – added to statistical views

Agenda

- Why?
- Approaches & Options
- Modernizing Database Definitions
- Modernizing Data Access
- Next Steps
- Conclusion Why SQL

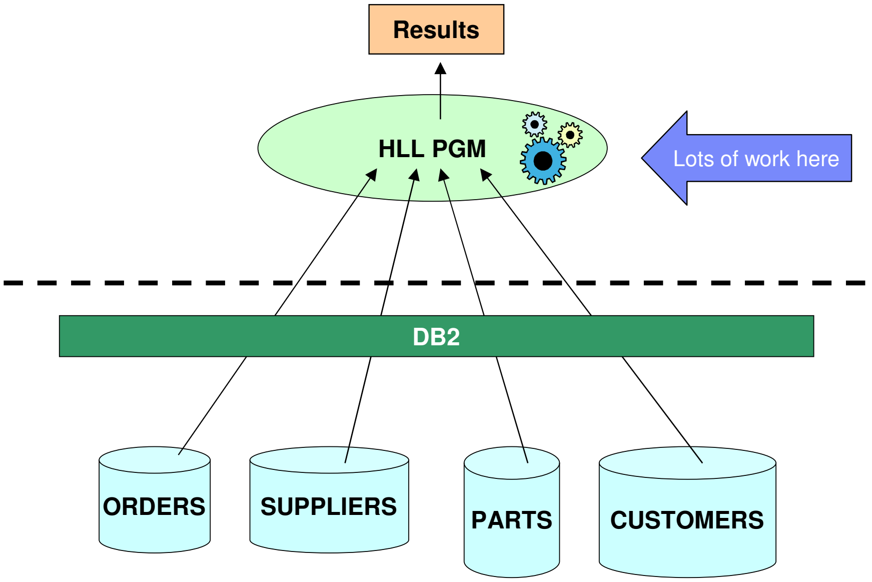
Why SQL?

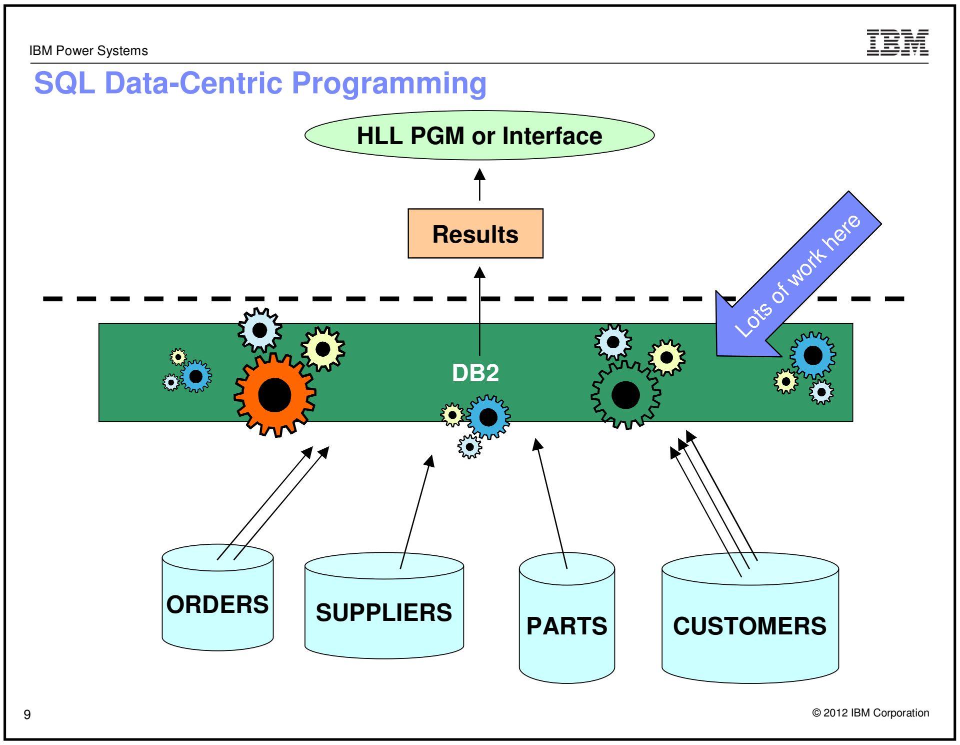
- Strategic database interface for industry
- Portability of code & skills
- Strategic interface for IBM i
- Faster delivery on IT requirements
- Performance & Scalability
- Increased Data Integrity
- Flexibility
- **Image – modern and DB server**

Want More Details…
NEW White Paper on Benefits of Modernizing with SQL
ibm.com/partnerworld/wps/servlet/ContentHandler/whitepaper/ibmi/db2/sql

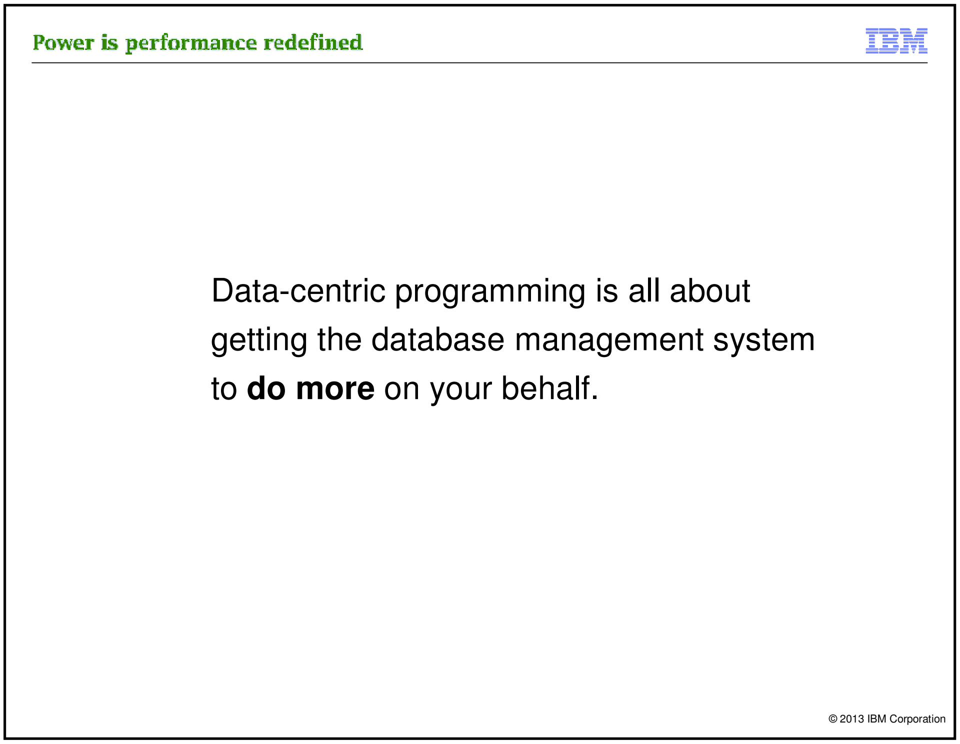
Why Data Centric ?

Traditional Record-Level Access

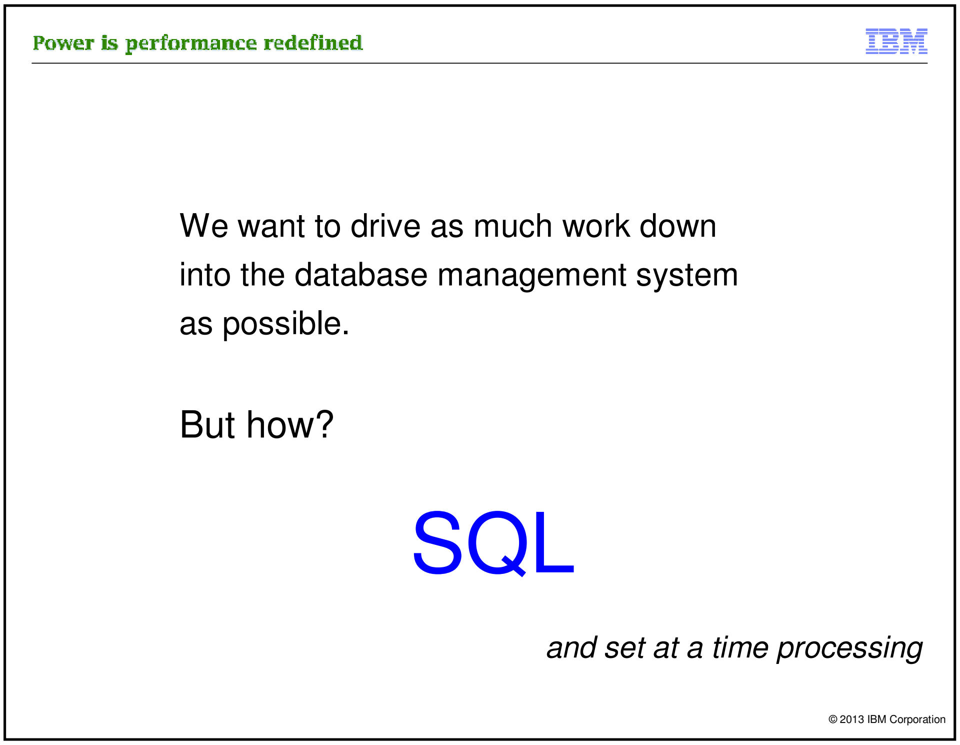




SQL Data-Centric Programming



Data-centric programming is all about getting the database management system to **do more** on your behalf.

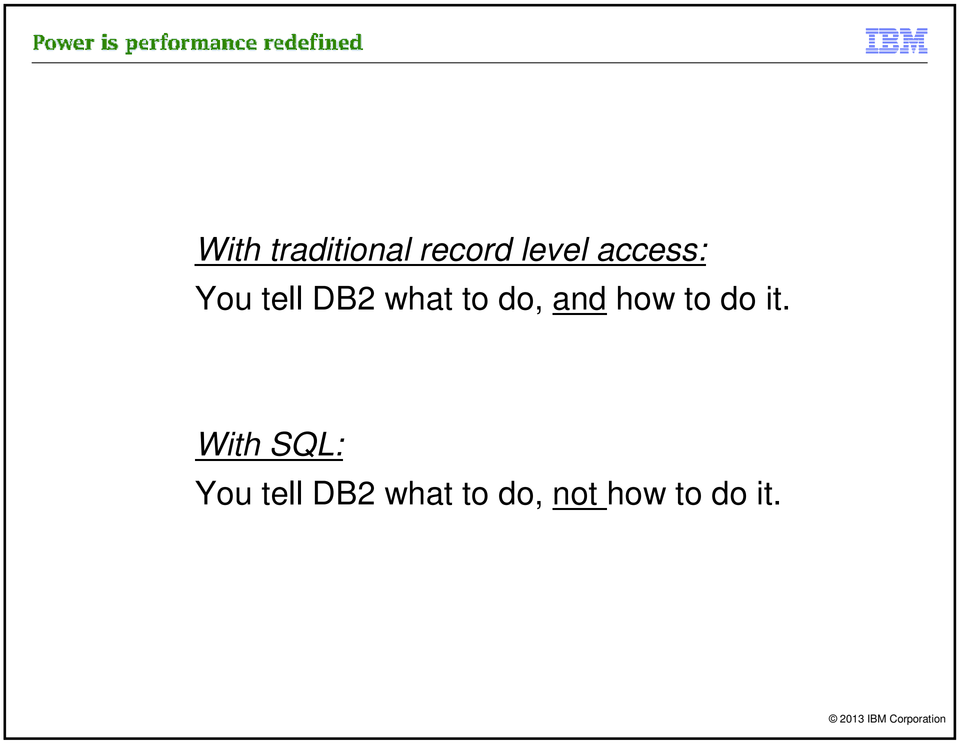


We want to drive as much work down into the database management system as possible.

But how?

SQL

and set at a time processing



With traditional record level access:

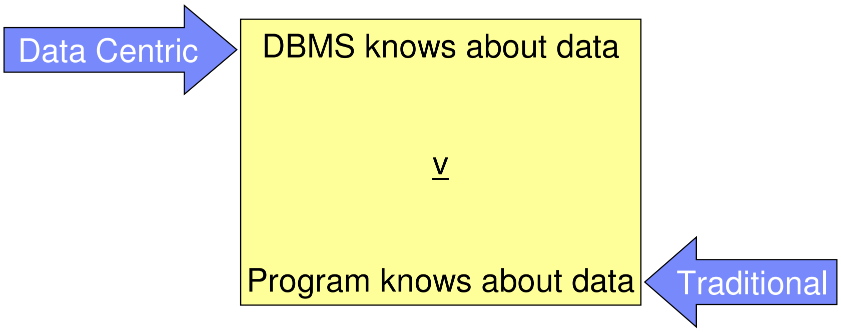
You tell DB2 what to do, and how to do it.

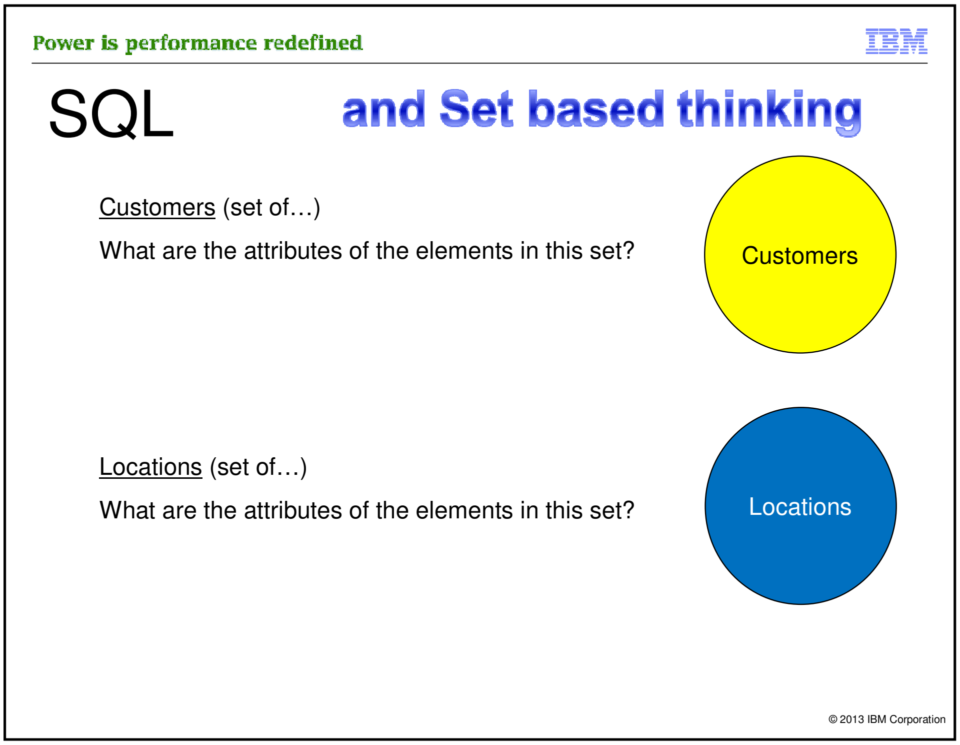
With SQL:

You tell DB2 what to do, not how to do it.

Set at a time processing ?

Enabling Sets





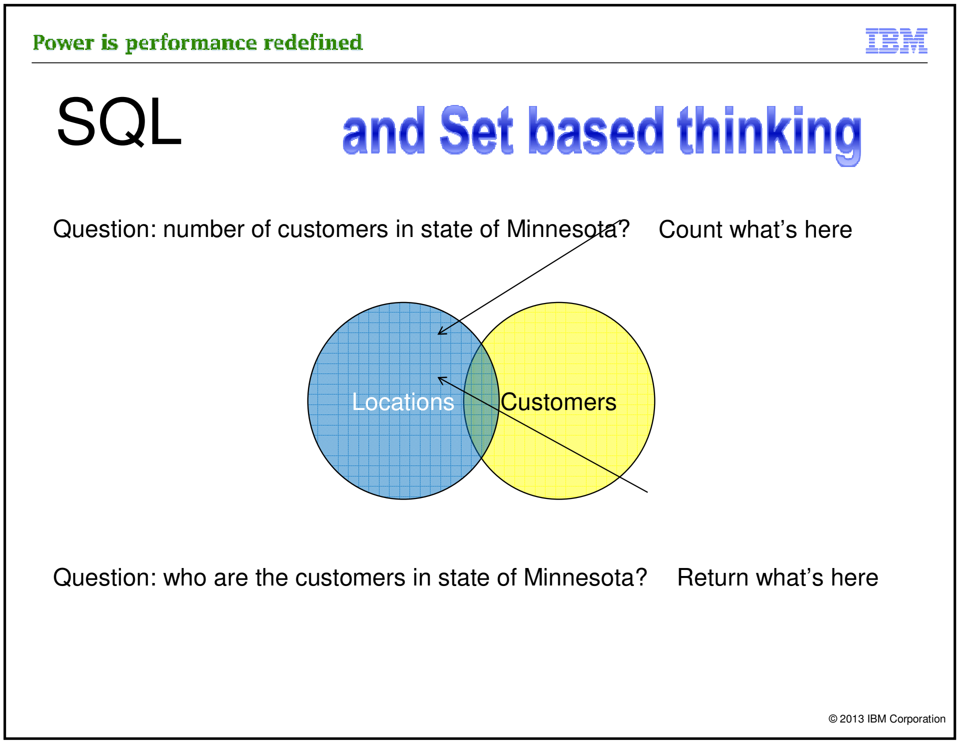
SQL and Set based thinking

Customers (set of…)

What are the attributes of the elements in this set?

Locations (set of…)

What are the attributes of the elements in this set?



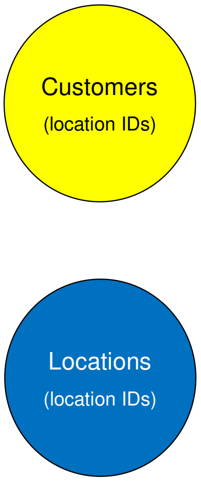
SQL and Set based thinking

Question: number of customers in state of Minnesota? Count what's here

Question: who are the customers in state of Minnesota? Return what's here

SQL and Set based thinking

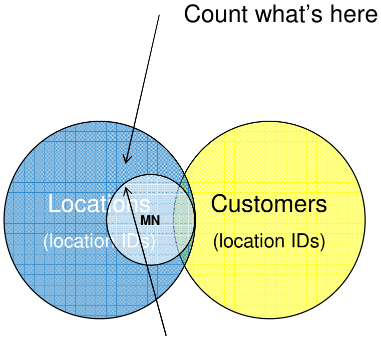
```
CREATE TABLE CUSTOMERS (
            CUSTOMER_NO INTEGER,
            CUSTOMER_NAME CHAR(50),
            LOCATION_ID INTEGER
)

CREATE TABLE LOCATIONS (
            LOCATION_ID INTEGER,
            CITY CHAR(50),
            COUNTY CHAR(50),
            STATE CHAR(2),
            ZIPCODE INTEGER
)
```



SQL and Set based thinking

```
SELECT COUNT(*)
FROM CUSTOMERS C
WHERE C.LOCATION_ID IN (
            SELECT L.LOCATION_ID
            FROM LOCATIONS L
            WHERE L.STATE = ‘MN’)

SELECT C.*
FROM CUSTOMERS C
WHERE C.LOCATION_ID IN (
            SELECT L.LOCATION_ID
            FROM LOCATIONS L
            WHERE L.STATE = ‘MN’)
```

Count what’s here



Return what’s here

WARNING: Record level access via SQL

The use of SQL statements does not necessarily mean "set at a time".

Traditional record level access processing can be done using SQL statements

DECLARE CURSOR

OPEN cursor

FETCH from cursor

CLOSE cursor

Read 1 record

Replacing record level access operations with SQL is NOT recommended!

Try to avoid reading and processing one unit of data at a time. **Think in sets.**

A Real and Recent Example of Misunderstanding

Sometimes you can use a sub-select that produces the same result. Granted this is NOT a join but with SQL there's never just one way...Lots of ways. Be sure to check your statement's performance with SQL Explain as some methods may perform better than others.

The example below would "join" all matching h.orderkey with any sub-selected d.orderkey first... Then return the single order 123456 from that set.

However switch the "where" and the "and" operands, and then you first select the single h.orderkey 123456 from orderheaders h, then the sub-select returns only the matching row from orderlines d. **Obviously performance would be better with the latter.**

```
-- A
select * from orderheaders h
where (h.orderkey = (select d.orderkey from orderlines d))
and (h.orderkey = 123456);

-- B
select * from orderheaders h
where (h.orderkey = 123456)
and (h.orderkey = (select d.orderkey from orderlines d));
```

A Real and Recent Example of Misunderstanding

```
--A
select * from orderheaders h
where (h.orderkey = (select d.orderkey from orderlines d))
and (h.orderkey = 123456);
```

So, the same, or different?

```
--B
select * from orderheaders h
where (h.orderkey = 123456)
and (h.orderkey = (select d.orderkey from orderlines d));
```

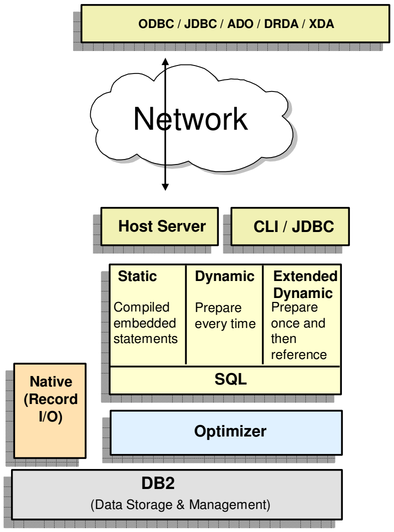
aHA!

These requests are essentially the same,

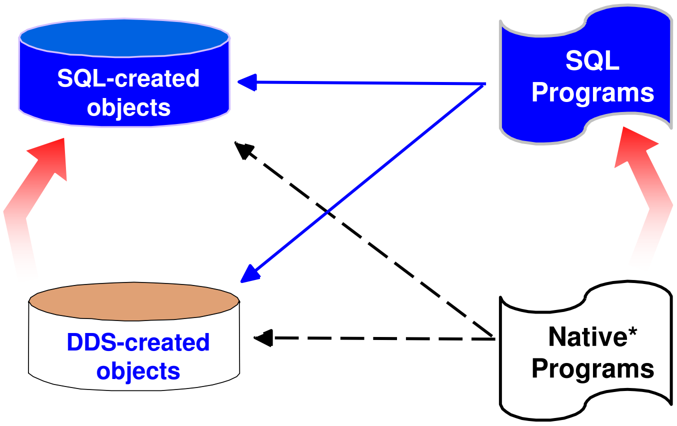
and DB2 for i optimizes them the same.

Think: SET at a time, not implementation

Approaches & Options


ODBC / JDBC / ADO / DRDA / XDA
Network
Host Server
CLI / JDBC
Static
Compiled embedded statements
Dynamic
Prepare every time
Extended Dynamic
Prepare once and then reference
SQL
Native (Record I/O)
Optimizer
DB2
(Data Storage & Management)

Approaches & Options



***Considerations*:**
Multi-member &
multi-format files

****Restrictions:***
EVIs, LOB columns,
UDTs, Datalinks, etc

Modernizing Definitions & Objects

- Modeling
- Terminology
- Moving from DDS to SQL DDL
- SQL object management
- Embedding business logic into database definitions

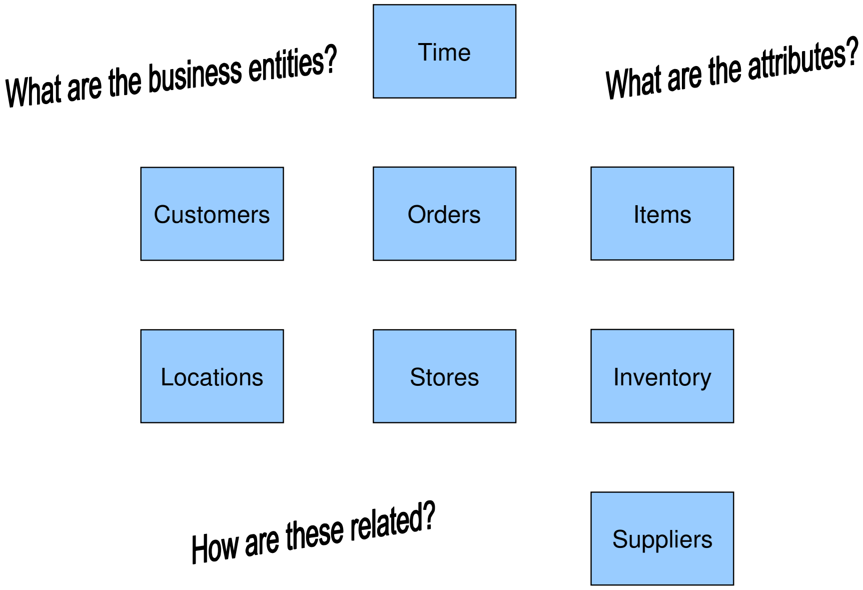
Data Modeling Concepts and Best Practices

Data Modeling

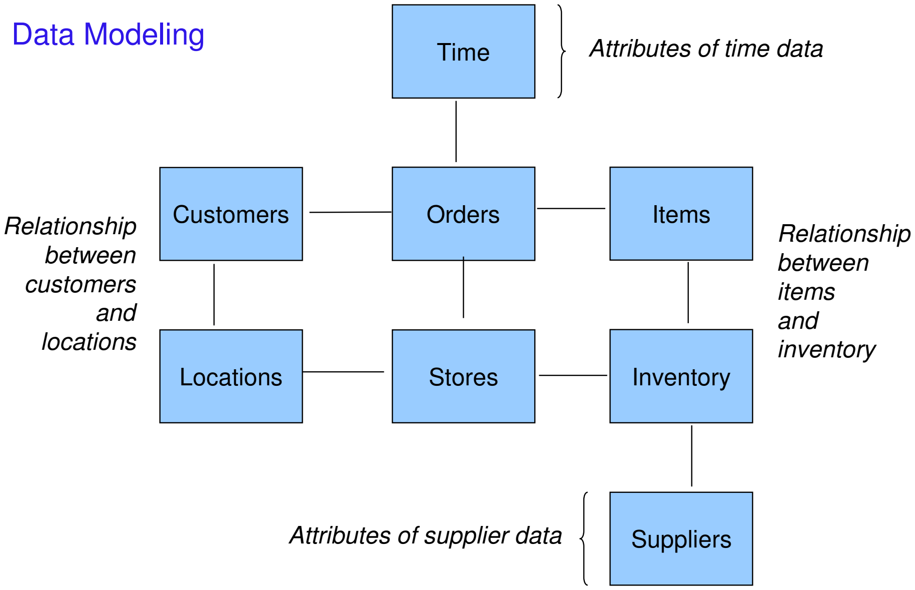
- Data modeling is a ***method*** used to define and analyze data requirements needed to support the business processes of an organization
- Data modeling is used to ***communicate*** the business rules and processes
- Data modeling is the ***process*** of creating a blueprint to visually represent data, its organization and the relationships between structures

More information:

www.information-management.com/issues/20_7/why-do-we-model-10019106-1.html?



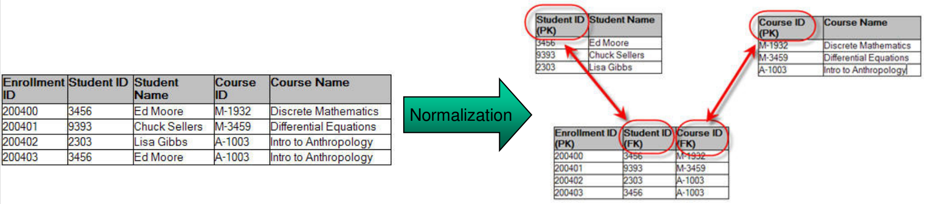
Data Modeling


Time
Attributes of time data
Relationship between customers and locations
Customers
Orders
Items
Relationship between items and inventory
Locations
Stores
Inventory
Attributes of supplier data
Suppliers

Modernizing Definitions & Objects

Data modeling

- "Master data" concept
  - –Services created to retrieve data – what if multiple copies exist?
- Database normalization
  - –Define a separate table for each related set of values
  - –Define the primary key (surrogate or natural)
  - –Eliminate redundant data
  - –Design for Fifth normal form (5NF), performance & storage may drop back to 3NF
  - –Establish RI constraints – increase data integrity
- A model facilitates communication and can provide Impact analysis for changes



Enrollment ID	Student ID	Student Name	Course ID	Course Name
200400	3456	Ed Moore	M-1932	Discrete Mathematics
200401	9393	Chuck Sellers	M-3459	Differential Equations
200402	2303	Lisa Gibbs	A-1003	Intro to Anthropology
200403	3456	Ed Moore	A-1003	Intro to Anthropology

Student ID (PK)	Student Name
3456	Ed Moore
9393	Chuck Sellers
2303	Lisa Gibbs

Course ID (PK)	Course Name
M-1932	Discrete Mathematics
M-3459	Differential Equations
A-1003	Intro to Anthropology

Enrollment ID (PK)	Student ID (FK)	Course ID (FK)
200400	3456	M-1932
200401	9393	M-3459
200402	2303	A-1003
200403	3456	A-1003

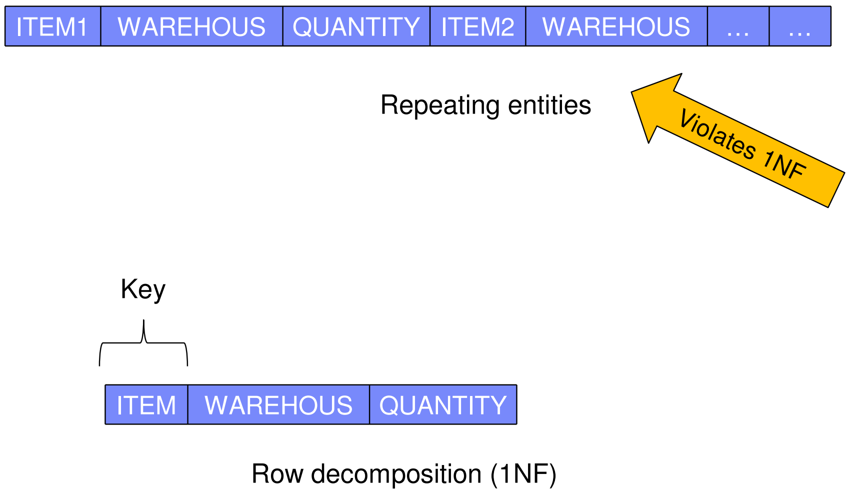
Normalization and Forms

- First Normal Form (1NF)
  - No duplicate rows (each row has a key of some type)
  - Eliminate duplicate columns from the same table – no arrays
  - Create separate tables for each group of related data
- Second Normal Form (2NF)
  - Meet all the requirements of 1NF
  - Remove subsets of data that apply to multiple rows and place in separate tables
  - Identify each row with a unique column
  - Create relationships between tables through the use of foreign keys
- Third Normal Form (3NF)
  - Meet all the requirements of 2NF
  - Remove columns that are not directly dependent upon the primary key

Usually Good enough

- Fourth Normal Form (4NF)
  - Meet all the requirements of 3NF
  - A relation is in 4NF if it has no multi-valued dependencies
- Fifth Normal Form (5NF)
  - *Makes my head hurt!*

Normalization and Forms (1NF)



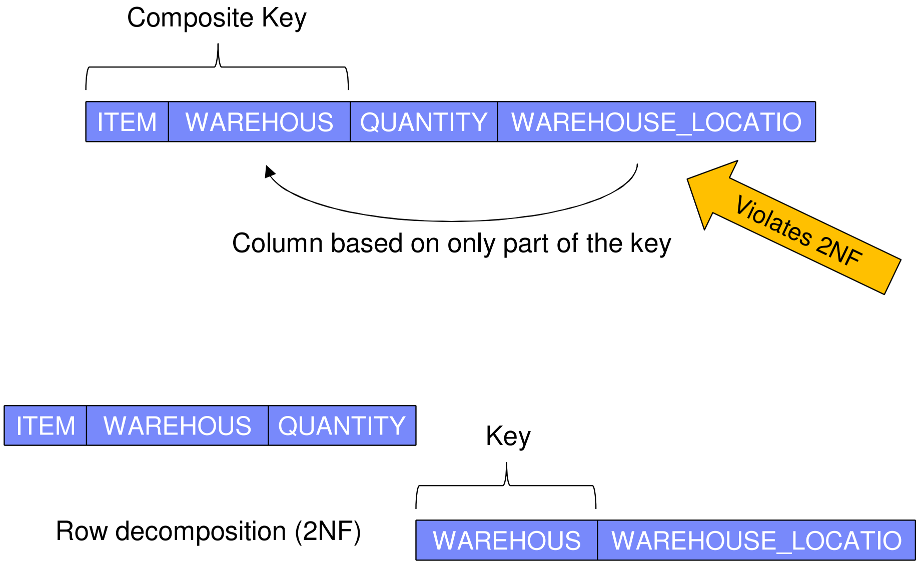
ITEM1	WAREHOUS	QUANTITY	ITEM2	WAREHOUS	…	…

Repeating entities

ITEM	WAREHOUS	QUANTITY

Row decomposition (1NF)

Normalization and Forms (2NF)



ITEM	WAREHOUS	QUANTITY	WAREHOUSE_LOCATIO

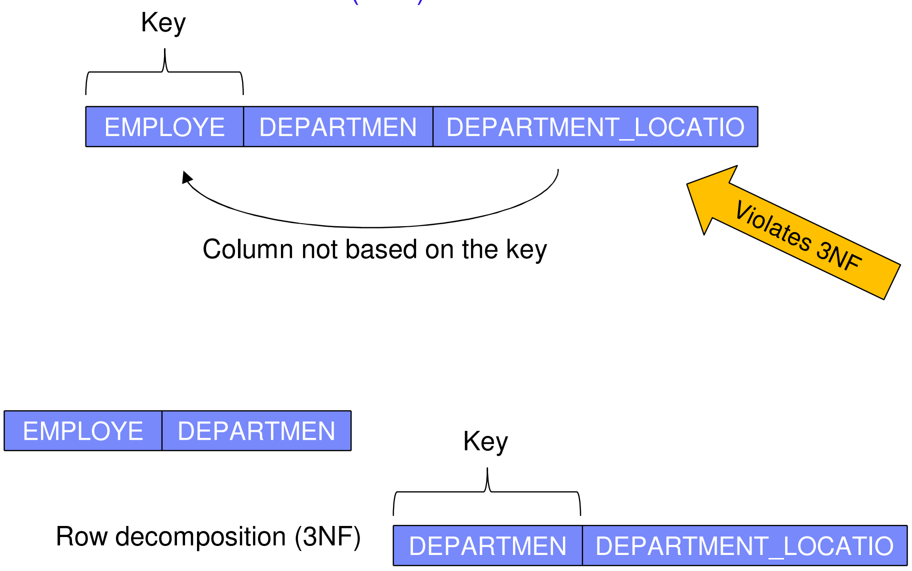
Column based on only part of the key

ITEM	WAREHOUS	QUANTITY

WAREHOUS	WAREHOUSE_LOCATIO

Row decomposition (2NF)

Normalization and Forms (3NF)



Key

EMPLOYE | DEPARTMEN | DEPARTMENT_LOCATIO

Column not based on the key

Violates 3NF

EMPLOYE | DEPARTMEN

Key

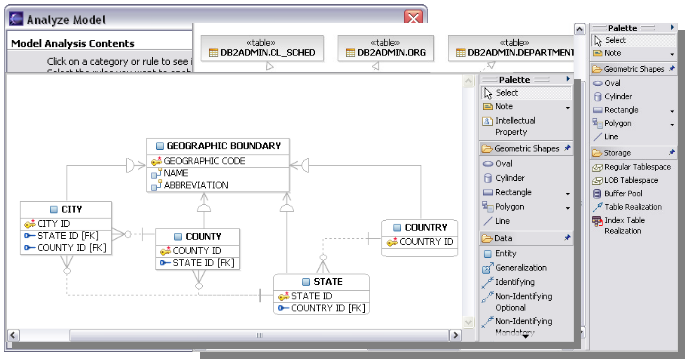
Row decomposition (3NF)

DEPARTMEN | DEPARTMENT_LOCATIO

Modernizing Definitions & Objects

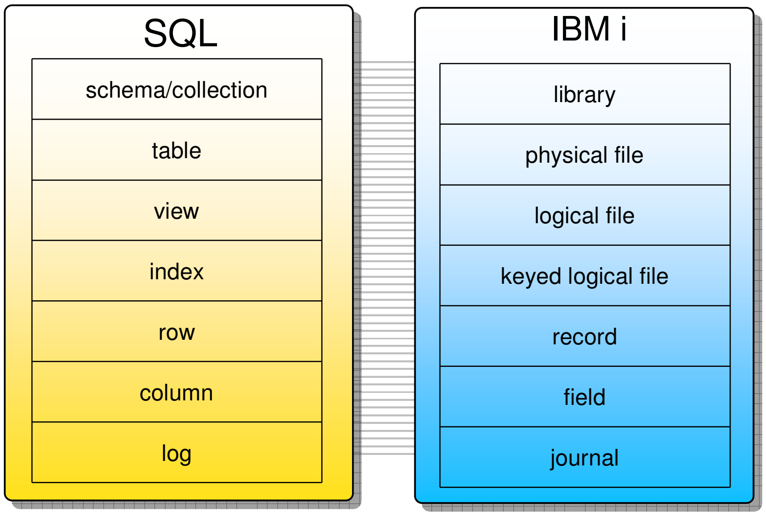
Data Modeling - IBM InfoSphere Data Architect (Version 7)

- Enterprise data modeling and management
  - Compare & synchronize
  - Forward & **reverse engineering**
  - Logical file support – Fixpack 003
  - Model analyzer for enterprise standard conformance
- Database development – SQL Stored Procedures and Function
- Trial Download: ibm.com/software/data/integration/rda/



Modernizing Database Objects

Terminology



SQL	IBM i
schema/collection	library
table	physical file
view	logical file
index	keyed logical file
row	record
column	field
log	journal

Moving from DDS to DDL objects

Modernizing Objects: CREATE TABLE vs CRTPF

```
CREATE TABLE EMP_MAST (
   EMP_MAST_PK  BIGINT
                  GENERATED BY DEFAULT AS IDENTITY
                  IMPLICITLY HIDDEN
                  PRIMARY KEY,
   EMPNO CHAR(6) UNIQUE,
   FIRSTNME VARCHAR(12),
   MIDINIT CHAR(1),
   LASTNAME VARCHAR(15),
   EMP_PICTURE BLOB(102400) ,
   EM_ROW_CHANGE_TS TIMESTAMP NOT NULL
      FOR EACH ROW ON UPDATE
         AS ROW CHANGE TIMESTAMP
      IMPLICITLY HIDDEN)
```

```
CRTPF FILE(EMPLOYEE) SRCFILE(QDDSSRC)
          SRCMBR(EMPLOYEE)
--Source Data
   A                                UNIQUE
   A         R EMPLOYEE
   A            EMPNO         6
   A            FIRSTNME     12   VARLEN
   A            MIDINIT       1
   A            LASTNAME     15  VARLEN
   A         K EMPNO


ADDPFCST FILE(EMPLOYEE) TYPE(*PRIKEY)
   KEY(EMPNO)
```

Wider selection of data types & column attributes

Longer, more descriptive identifiers

Data modeling tool support

Self-contained source statement, can include constraint definitions

Limited set of data types & attributes

Format sharing & field attributes (CHECK, RANGE, DATFMT)

Keyed support, but only 1 key per definition.

Constraints must be defined separately

Modernizing Objects: CREATE INDEX vs CRTLF (**Keyed**)

```
CREATE INDEX EMP_LASTNAME_DEPT
  ON EMP_MAST(WORKDEPT, LASTNAME)
   RCDFMT EMPLOYEER1
   ADD COLUMNS
           EMPNO, FIRSTNME, MIDINIT

CREATE ENCODED VECTOR INDEX RegionIX
   ON SALES(REGION)
CREATE ENCODED VECTOR INDEX RegionIXAgg
   ON SALES(REGION) INCLUDE(SUM(SALESREV))
```

```
CRTLF FILE(EMPLOYEEL1) SRCFILE(QDDSSRC)
   SRCMBR(EMPLOYEEL1)
--Source Data
A         R EMPLOYEER1    PFILE(EMPLOYEE)
A            WORKDEPT
A            LASTNAME
A            EMPNO
A            FIRSTNME
A            MIDINIT
A         K WORKDEPT
A         K LASTNAME
```

Encoded Vector Index (EVI) structure

Expressions can be used in the definition of the key columns

Sparse Indexes with WHERE clause (ie, Select/Omit)

EVI "Instant" Aggregate support

Larger default logical page size

Only Binary Radix Tree structure support – no EVIs

Limited support for key derivations and expressions

Key attributes – ALTSEQ, DIGIT, FCFO, FIFO, LIFO, UNSIGNED, ZONE

Smaller default logical page size

Modernizing Objects: CREATE VIEW vs CRTLF (non-keyed)

```
CREATE VIEW
   EMPLOYEE_BONUSES_BY_DEPARTMENT_WITH
   IN_STATE
AS
SELECT EA.STATE, DM.DEPTNAME,
   SUM(EM.BONUS)
FROM EMAST EM
 JOIN EADDR EA USING (EM_PK)
 JOIN DMAST DM ON WRKDPT = DPTNO
GROUP BY EA.STATE, DM.DEPTNAME
```

```
CRTLF FILE(EMPLOYEEJ1) SRCFILE(QDDSSRC)
   SRCMBR(EMPLOYEEJ1)
--Source Data
A      R EMPLOYEEJA  JFILE(EMAST  EADDR +
A                              DMAST)
A      J                    JOIN(1 2)
A                           JFLD(EM_PK EM_PK)
A      J                    JOIN(1 3)
A                           JFLD(WRKDPT DPTNO)
A        STATE
A        DEPTNAME
A        BONUS
```

Full access to advanced query capabilities of SQL

Can be used as logical files to enhance native functionality

No support for keying/ordering

Limited Join support

No support for Grouping, Case, Subqueries, User-Defined functions, …

Multiple members & formats

Modernizing Objects: CREATE VIEW vs CRTLF

- SQL Views cannot be keyed/ordered… does that mean they are slower than Logical Files?
  - NO - assuming you have the right set of indexes/statistics in place for the query optimizer to use
  - View is used by SQL just to transform data, query optimizer's job to find the best method to speed up selection or sorting
  - Fastest method **may not be a keyed access** method

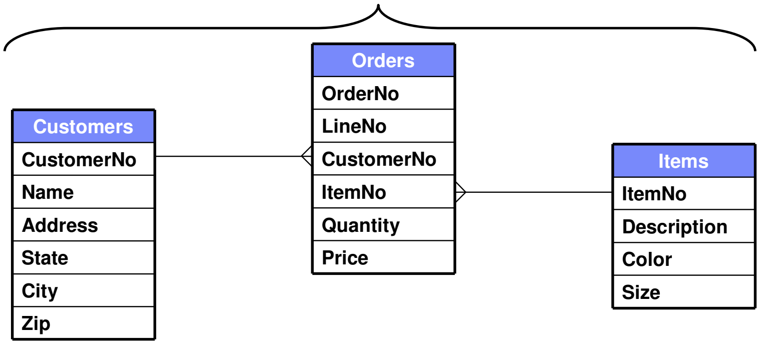
Views

- **Appear like tables to an application program or query interface**
- **Contain no data**
- ***Logically* represents one or more tables over which they are created**
- **Can represent all the columns and all of the rows of the given tables or a subset of them**
  - The columns can be arranged differently in a view than they are in the tables from which they are taken
- **Represent no order of rows**
  - An ORDER BY clause specifies the final order of the result set
- **Can be used to simplify and insulate the underlying data model**
  - Hide local selection, joins, grouping, etc.
  - Only views are directly referenced, never the tables

Views

```
SELECT *
FROM VIEW1
WHERE CUSTOMERNO = 112358
ORDER BY ITEMNO;
```

```
CREATE VIEW VIEW1(customerno, name, itemno, sumqty) as
(SELECT C.CUSTOMERNO, C.NAME, I.ITEMNO,
SUM(O.QUANTITY)
FROM CUSTOMERS C
INNER JOIN ORDERS O ON C.CUSTOMERNO = O.CUSTOMERNO
LEFT OUTER JOIN ITEMS I ON O.ITEMNO = I.ITEMNO
```



Customers
CustomerNo
Name
Address
State
City
Zip

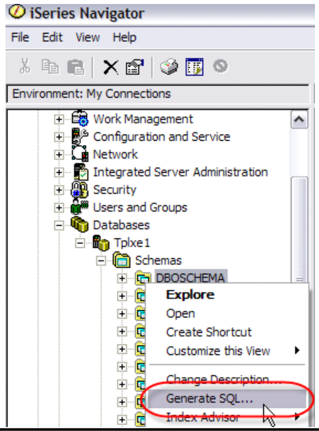
Orders
OrderNo
LineNo
CustomerNo
ItemNo
Quantity
Price

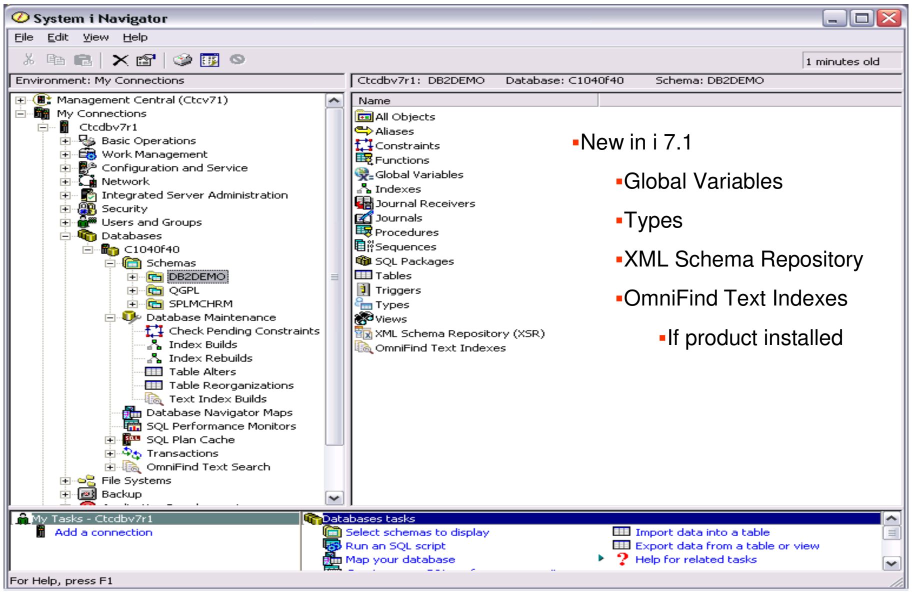
Items
ItemNo
Description
Color
Size

Modernizing Database Definitions & Objects

DDS to SQL Conversion Tool

- System i Navigator Generate SQL Task (QSQGNDDL API)
  - Useful in converting object definitions from DDS to SQL
  - Supports physical & logical files
    - Not all DDS features can be converted, tool will convert as much as possible and **generate warnings** for unconvertible options (e.g., EDTCDE)
    - Logical files converted to SQL Views
      - » **API can do index equivalent**
    - SQL Field Reference File support not used
  - Can convert a single object or a group of objects
  - Output can be edited & saved directly into source file members
  - Tip you can generate DDL for any existing SQL object



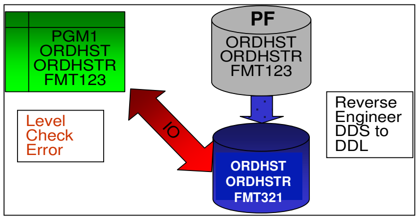


- New in i 7.1
  - Global Variables
  - Types
  - XML Schema Repository
  - OmniFind Text Indexes
    - If product installed

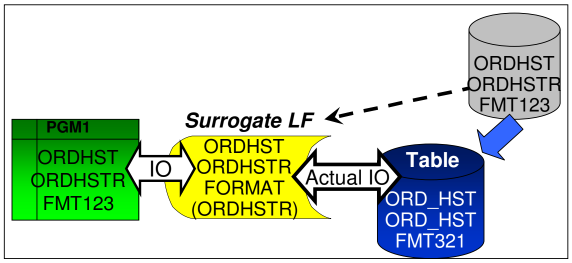
Power your planet.

Modernizing Database Definitions - Transparently

- Converting DDS PF to SQL DDL Table results in format identifiers being changed
  - HLL programs accessing the SQL Table will receive a “level check” exception message.
  - Only solutions prior to 5.4
    - recompile the program or
    - ignore the exception (not recommended)

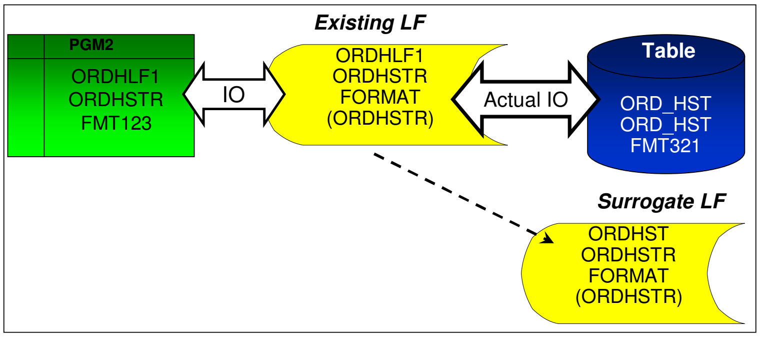


- A **surrogate file** preserves the original DDS PF format
  - Allows new columns to be added to SQL DDL Table
  - FORMAT keyword used to share surrogate format
    - Prevents level check IDs for programs accessing original PF or LFs sharing format
- “Best” method for avoiding format id changes!



Modernizing Database Definitions – Transparently

- Logical files also need to re-engineered to reference the SQL table
  - For each logical file which **shared** the physical file format (FMT123):
    - PFILE modified to point at SQL table (FMT321)
    - FORMAT keyword specifies surrogate LF (FMT123)
  - Some LFs don’t require re-engineering
    - DDS LF with unique format name
    - DDS Join Logical Files have unique format IDs



Modernizing Database Definitions - Transparently

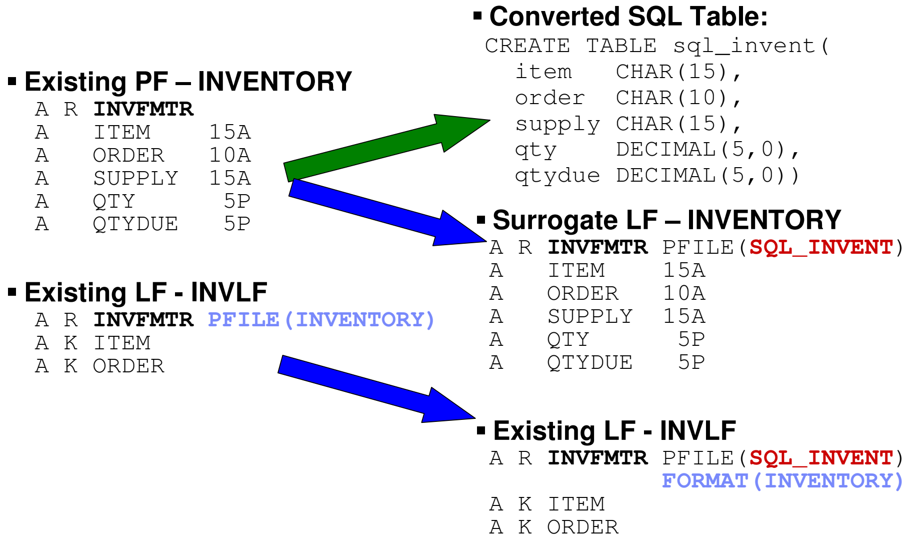
1. Convert PF to SQL Table (with new name)

2. Create SQL indexes to replace any implicitly created keyed access paths that exist for DDS files (use “Show Indexes”)
   - Why? here

3. Create “Surrogate” LF with same name as original PF name

4. Modify existing LFs to reference SQL table

Transparent SQL Migration - Example



- **Existing PF – INVENTORY**

```
A R INVFMTR
A   ITEM     15A
A   ORDER    10A
A   SUPPLY   15A
A   QTY       5P
A   QTYDUE    5P
```

- **Existing LF - INVLF**

```
A R INVFMTR PFILE(INVENTORY)
A K ITEM
A K ORDER
```

- **Converted SQL Table:**

```
CREATE TABLE sql_invent(
  item   CHAR(15),
  order  CHAR(10),
  supply CHAR(15),
  qty    DECIMAL(5,0),
  qtydue DECIMAL(5,0))
```

- **Surrogate LF – INVENTORY**

```
A R INVFMTR PFILE(SQL_INVENT)
A   ITEM     15A
A   ORDER    10A
A   SUPPLY   15A
A   QTY       5P
A   QTYDUE    5P
```

- **Existing LF - INVLF**

```
A R INVFMTR PFILE(SQL_INVENT)
            FORMAT(INVENTORY)
A K ITEM
A K ORDER
```

SQL object managment

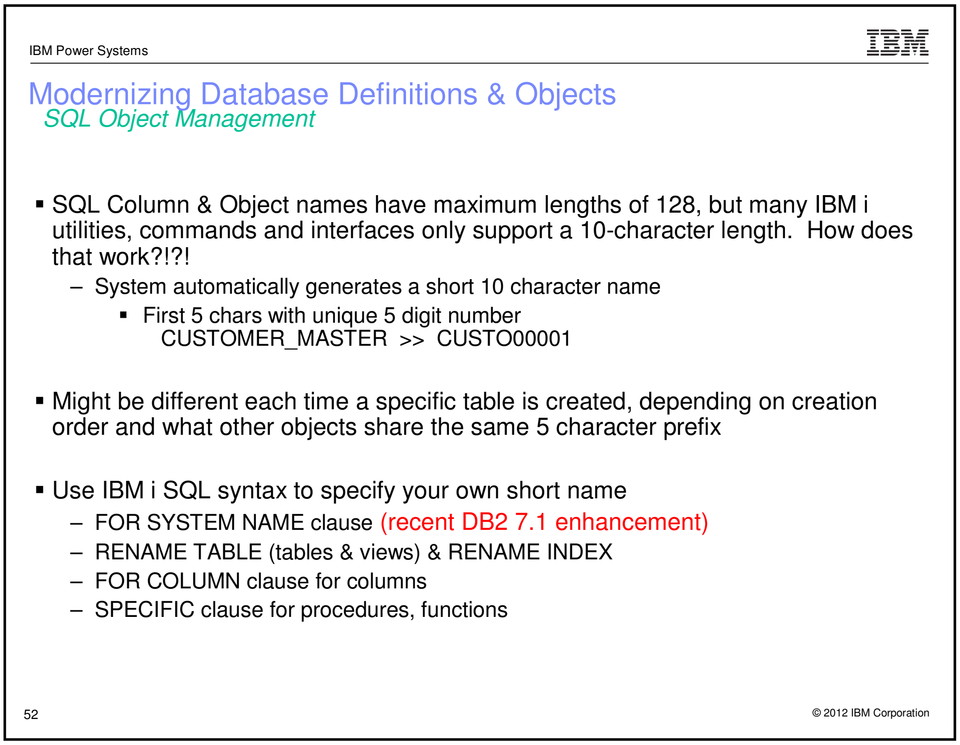
Modernizing Database Definitions & Objects

SQL Object management

- SQL Source Management best practices:
  - Just like DDS, SQL source can be stored in source physical file members just and referenced with the RUNSQLSTM CL command instead of CRTPF/CRTLF
    - If change management tools are not IBM i specific, store SQL scripts in IFS or PC
    - If SQL source misplaced, Generate SQL can be used to retrieve the SQL source from System Catalogs (SYSIBM & QSYS2)
    - Navigator Run SQL Scripts in 6.1 can store and retrieve SQL from source members
  - SQL Table definitions can use Field Reference File – **CREATE TABLE LIKE**

```
 CREATE TABLE customer AS
  (SELECT id cust_id, lname cust_lastname, fname cust_firstname,
      city cust_city FROM RefFile)
  WITH NO DATA
```

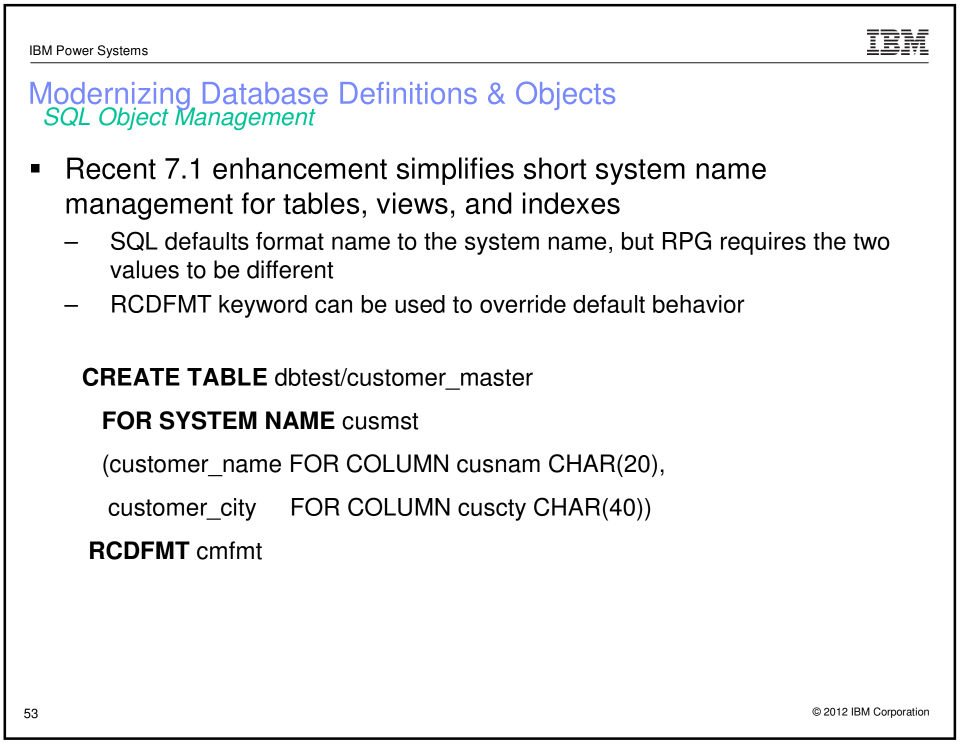
- May need to adjust process for moving from development to production
  - Best practice is to re-execute SQL creation script
  - Save/Restore process for SQL databases documented at: ibm.com/developerworks/db2/library/techarticle/0305milligan/0305milligan.html



Modernizing Database Definitions & Objects

SQL Object Management

- SQL Column & Object names have maximum lengths of 128, but many IBM i utilities, commands and interfaces only support a 10-character length. How does that work?!?!
  - System automatically generates a short 10 character name
    - First 5 chars with unique 5 digit number
      CUSTOMER_MASTER >> CUSTO00001
- Might be different each time a specific table is created, depending on creation order and what other objects share the same 5 character prefix
- Use IBM i SQL syntax to specify your own short name
  - FOR SYSTEM NAME clause (recent DB2 7.1 enhancement)
  - RENAME TABLE (tables & views) & RENAME INDEX
  - FOR COLUMN clause for columns
  - SPECIFIC clause for procedures, functions



Modernizing Database Definitions & Objects

SQL Object Management

- Recent 7.1 enhancement simplifies short system name management for tables, views, and indexes
  - SQL defaults format name to the system name, but RPG requires the two values to be different
  - RCDFMT keyword can be used to override default behavior

```
CREATE TABLE dbtest/customer_master
  FOR SYSTEM NAME cusmst
  (customer_name FOR COLUMN cusnam CHAR(20),
   customer_city     FOR COLUMN cuscty CHAR(40))
 RCDFMT cmfmt
```

Modernizing Database Definitions & Objects

SQL Object Management

Pre-7.1 solution

- Short & Long Name Co-existence Example
  - Specify the short name at creation:

```
CREATE TABLE dbtest/cusmst
 (customer_name FOR COLUMN cusnam CHAR(20),
  customer_city FOR COLUMN cuscty CHAR(40))
```

  - Specify a long name for existing short-name:

```
RENAME TABLE dbtest/cusmst TO customer_master
      FOR SYSTEM NAME cusmst
```

- If long name specified on SQL Table definition, can also add/control the short name after table created:

```
RENAME TABLE dbtest/customer_master TO SYSTEM NAME cusmst
```

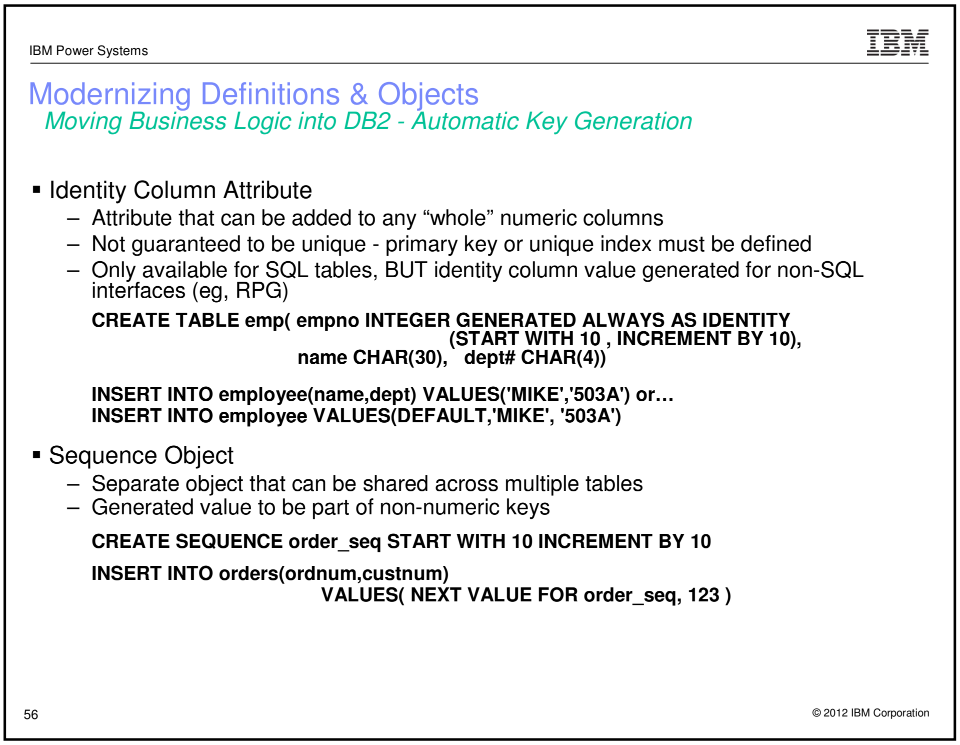
Modernizing Definitions & Objects

SQL & Non-relational data

- User-Defined Table Functions
  - Allows non-relational & legacy data to be virtualized as an SQL table

```
SELECT * FROM TABLE(myudtf('Part XYZ'))
```

  - Both SQL & External Table Functions supported
    - External UDTFs can be easily written to access multi-format files, S/36 files, and stream files
    - Table functions need to be invoked from SQL-based interfaces or SQL view
- LOBs
  - Allows you to keep non-relational data along with all the other business data
- Datalinks
  - URL-based data type to provide linkage to related objects in IFS
  - Can establish RI relationship between table row & IFS object
- HTTP services and XMLTable
  - .ibm.com/partnerworld/wps/servlet/ContentHandler/stg_ast_sys_wp_access_web_service_db2_i_udf



Modernizing Definitions & Objects

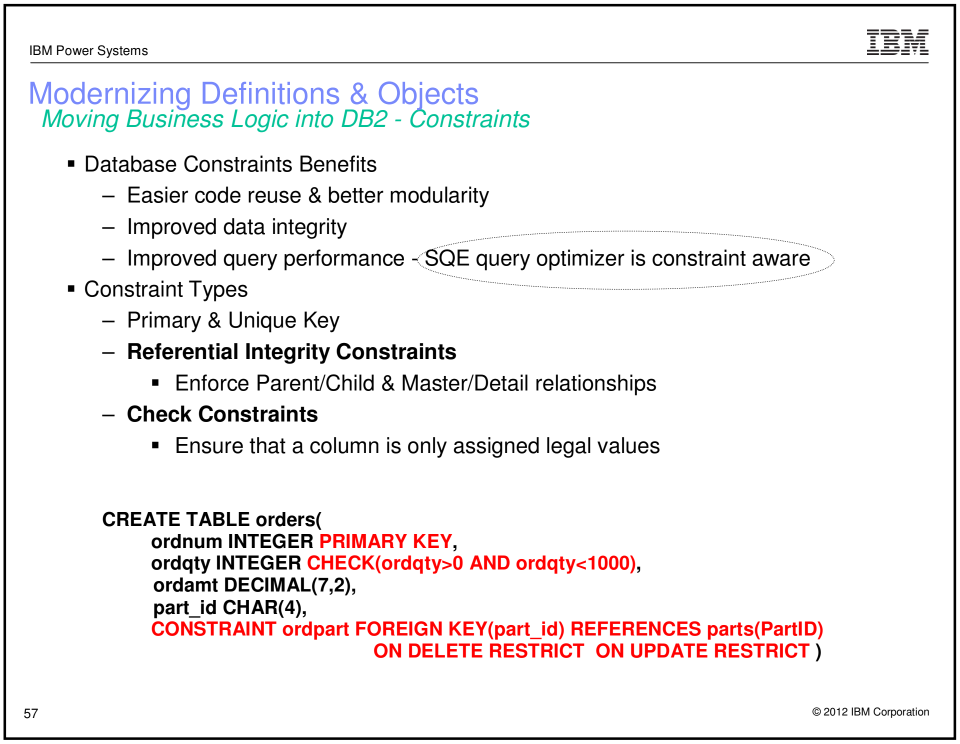
Moving Business Logic into DB2 - Automatic Key Generation

- Identity Column Attribute
  - Attribute that can be added to any “whole” numeric columns
  - Not guaranteed to be unique - primary key or unique index must be defined
  - Only available for SQL tables, BUT identity column value generated for non-SQL interfaces (eg, RPG)

```
CREATE TABLE emp( empno INTEGER GENERATED ALWAYS AS IDENTITY
                                         (START WITH 10 , INCREMENT BY 10),
                  name CHAR(30),   dept# CHAR(4))

INSERT INTO employee(name,dept) VALUES('MIKE','503A') or…
INSERT INTO employee VALUES(DEFAULT,'MIKE', '503A')
```

- Sequence Object
  - Separate object that can be shared across multiple tables
  - Generated value to be part of non-numeric keys

```
CREATE SEQUENCE order_seq START WITH 10 INCREMENT BY 10

INSERT INTO orders(ordnum,custnum)
                      VALUES( NEXT VALUE FOR order_seq, 123 )
```



Modernizing Definitions & Objects

Moving Business Logic into DB2 - Constraints

- Database Constraints Benefits
  - Easier code reuse & better modularity
  - Improved data integrity
  - Improved query performance - SQE query optimizer is constraint aware
- Constraint Types
  - Primary & Unique Key
  - **Referential Integrity Constraints**
    - Enforce Parent/Child & Master/Detail relationships
  - **Check Constraints**
    - Ensure that a column is only assigned legal values

```
CREATE TABLE orders(
      ordnum INTEGER PRIMARY KEY,
      ordqty INTEGER CHECK(ordqty>0 AND ordqty<1000),
      ordamt DECIMAL(7,2),
      part_id CHAR(4),
      CONSTRAINT ordpart FOREIGN KEY(part_id) REFERENCES parts(PartID)
                               ON DELETE RESTRICT  ON UPDATE RESTRICT )
```

Constraints

- **Database constraints define business rules**
- **DB2 provides methods to enforce the rules**
  - Indexes are created to support the enforcement
- **Constraints can assist the query optimizer and DB engine**
  - Rules enforced by the DB2 provide the guarantees
  - Rules enforced by your programs do not
- **Example of data centric programming to minimize coding**
  - Let the DB2 server do the work!
- **AccessPaths used to enforce constraints are there for SQL access as well**

✓ **Unique key constraint**
✓ **Primary key constraint**
✓ **Referential constraint**
✓ **Check constraint**

Modernizing Definitions & Objects

Moving Business Logic into DB2 - Triggers

- Triggers allow you initiate business policies & processes whenever new data comes in or existing data is changed
  - DB2 responsible for always invoking the trigger program
  - Execution is independent of the user interface
  - Can be used to transform data before it gets into DB2
- DB2 for i Trigger Support
  - Before & After: Insert, Update, & Delete events (up to 300 triggers)
  - SQL & External(ADDPFTRG) Triggers
    - Column-level, Statement-level, and Instead Of triggers only available with SQL Triggers
    - **Multiple Event Triggers** (v7.1 TR6)

```
CREATE TRIGGER audit_salary
  AFTER UPDATE ON employee(salary)
  REFERENCING NEW AS n
  REFERENCING OLD AS o
  FOR EACH ROW
  WHEN (n.salary - o.salary >= 5000)
      INSERT INTO audit
            VALUES(n.empno, n.deptno, n.salary,current timestamp)
```

Modernizing Data Access

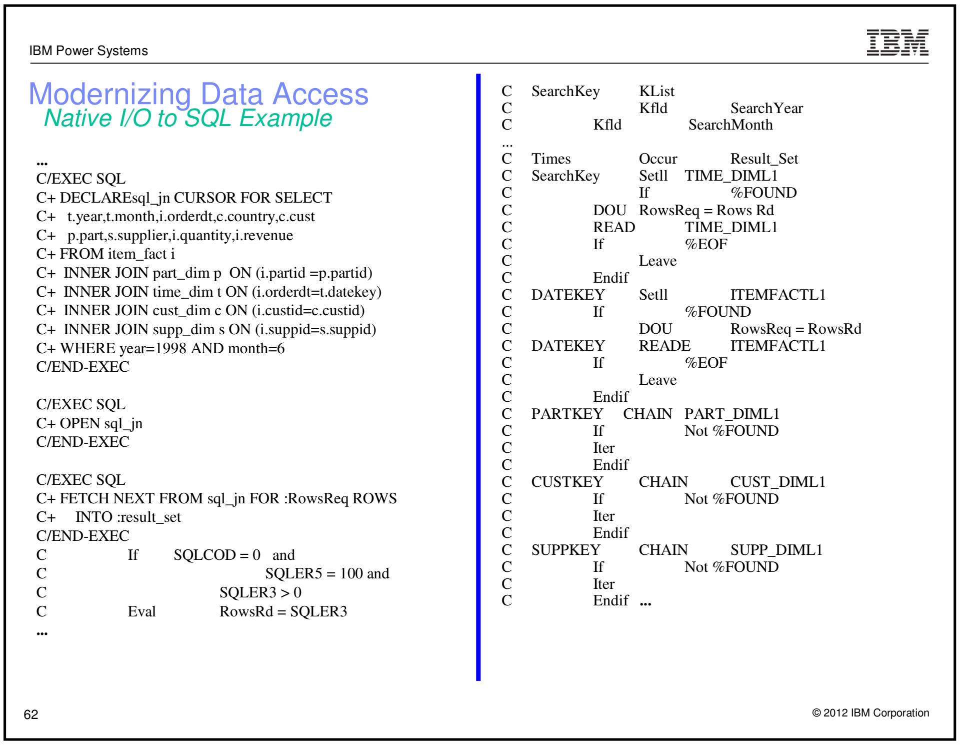
- Programming Interfaces

- Native I/O to SQL Comparison

Modernizing Data Access – Programming Interfaces

Static SQL	**Dynamic SQL**	**Extended Dynamic SQL**
Embedded Static	Embedded Dynamic	QSQPRCED
SQL Procedures, Functions, Triggers	SQL Procedures, Functions, Triggers	Toolbox JDBC driver
	JDBC, SQLJ	IBM i Access ODBC & OLE DB
	OLE DB, .NET	XDA APIs
	CLI, ODBC	
	PHP ibm_db2	
	RUNSQLSTM	

*****DB2 SQL Development Kit only required if embedded SQL (& STRSQL) is going to be used***

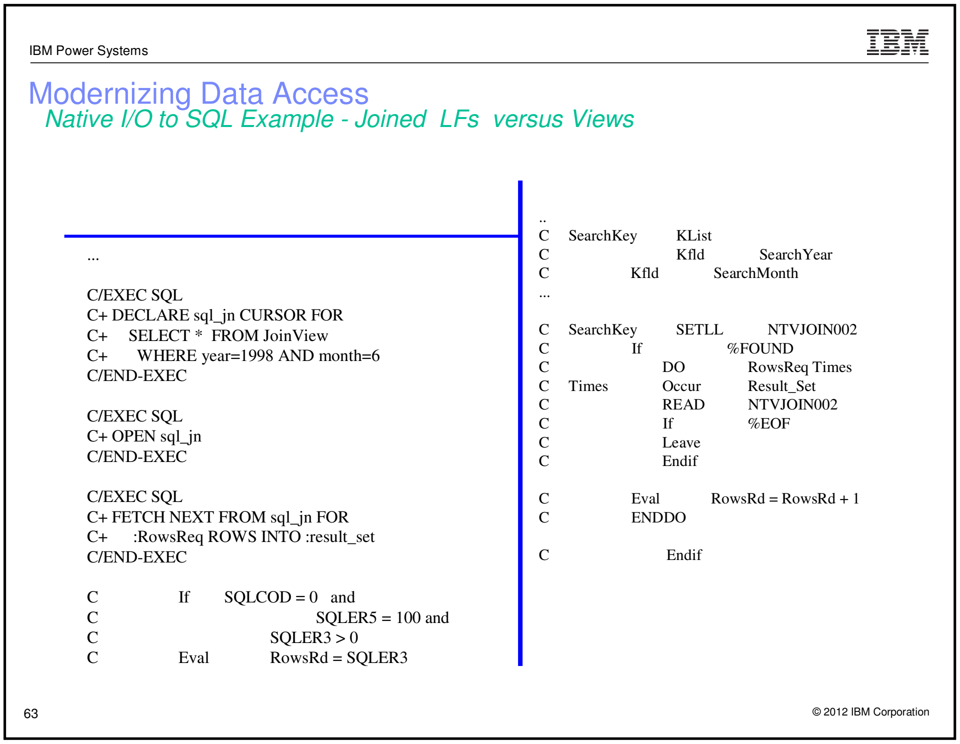


Modernizing Data Access

Native I/O to SQL Example

```
...
C/EXEC SQL
C+ DECLAREsql_jn CURSOR FOR SELECT
C+   t.year,t.month,i.orderdt,c.country,c.cust
C+   p.part,s.supplier,i.quantity,i.revenue
C+ FROM item_fact i
C+  INNER JOIN part_dim p  ON (i.partid =p.partid)
C+  INNER JOIN time_dim t ON (i.orderdt=t.datekey)
C+  INNER JOIN cust_dim c ON (i.custid=c.custid)
C+  INNER JOIN supp_dim s ON (i.suppid=s.suppid)
C+ WHERE year=1998 AND month=6
C/END-EXEC

C/EXEC SQL
C+ OPEN sql_jn
C/END-EXEC

C/EXEC SQL
C+ FETCH NEXT FROM sql_jn FOR :RowsReq ROWS
C+     INTO :result_set
C/END-EXEC
C                If       SQLCOD = 0   and
C                                        SQLER5 = 100 and
C                                SQLER3 > 0
C                Eval            RowsRd = SQLER3
...
```

```
C   SearchKey        KList
C                    Kfld             SearchYear
C           Kfld             SearchMonth
...
C   Times            Occur            Result_Set
C   SearchKey        Setll    TIME_DIML1
C                    If               %FOUND
C           DOU   RowsReq = Rows Rd
C           READ            TIME_DIML1
C           If              %EOF
C                    Leave
C           Endif
C   DATEKEY          Setll            ITEMFACTL1
C           If              %FOUND
C                    DOU              RowsReq = RowsRd
C   DATEKEY          READE            ITEMFACTL1
C           If              %EOF
C                    Leave
C           Endif
C   PARTKEY    CHAIN    PART_DIML1
C           If              Not %FOUND
C           Iter
C           Endif
C   CUSTKEY          CHAIN            CUST_DIML1
C           If              Not %FOUND
C           Iter
C           Endif
C   SUPPKEY          CHAIN            SUPP_DIML1
C           If              Not %FOUND
C           Iter
C           Endif  ...
```



Modernizing Data Access

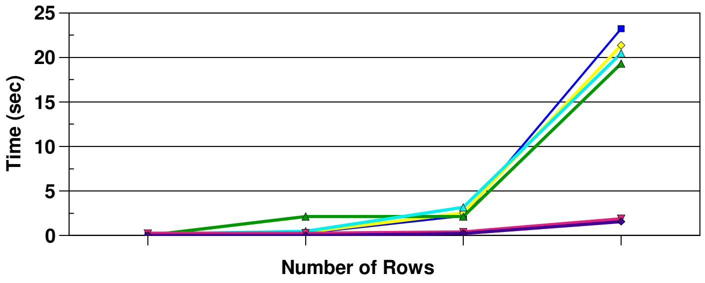
Native I/O to SQL Example - Joined LFs versus Views

```
...

C/EXEC SQL
C+ DECLARE sql_jn CURSOR FOR
C+     SELECT *  FROM JoinView
C+       WHERE year=1998 AND month=6
C/END-EXEC

C/EXEC SQL
C+ OPEN sql_jn
C/END-EXEC

C/EXEC SQL
C+ FETCH NEXT FROM sql_jn FOR
C+      :RowsReq ROWS INTO :result_set
C/END-EXEC

C                If       SQLCOD = 0   and
C                                         SQLER5 = 100 and
C                                 SQLER3 > 0
C                Eval            RowsRd = SQLER3
```

```
..
C   SearchKey        KList
C                    Kfld            SearchYear
C           Kfld             SearchMonth
...

C   SearchKey        SETLL           NTVJOIN002
C           If                %FOUND
C                DO             RowsReq Times
C   Times        Occur          Result_Set
C                READ           NTVJOIN002
C                If             %EOF
C                Leave
C                Endif

C           Eval          RowsRd = RowsRd + 1
C           ENDDO

C                Endif
```

Modernizing Data Access

Native I/O to SQL Example - Performance Comparison



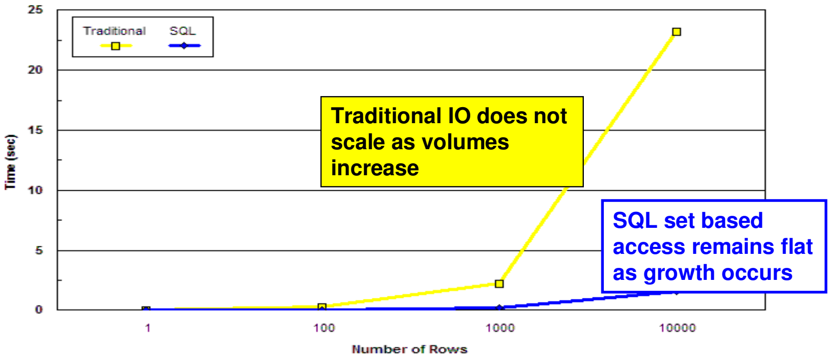
Note: Tests run on Model 720 w/1600 CPW & 2 GB Memory - your performance results may vary

	1	100	1000	10000
■ Native File Join	0.002512	0.260248	2.219504	23.228176
◇ Native JoinLF	0.002304	0.362128	2.544608	21.366480
▲ Native JoinLF w	0.002400	2.144288	2.125032	19.311464
△ SQL - No IOA	0.145160	0.489136	3.166704	20.452984
▼ SQL IOA	0.251168	0.267208	0.417800	1.898800
◆ SQL SQE IOA	0.013536	0.019320	0.250160	1.576536

Modernizing Data Access

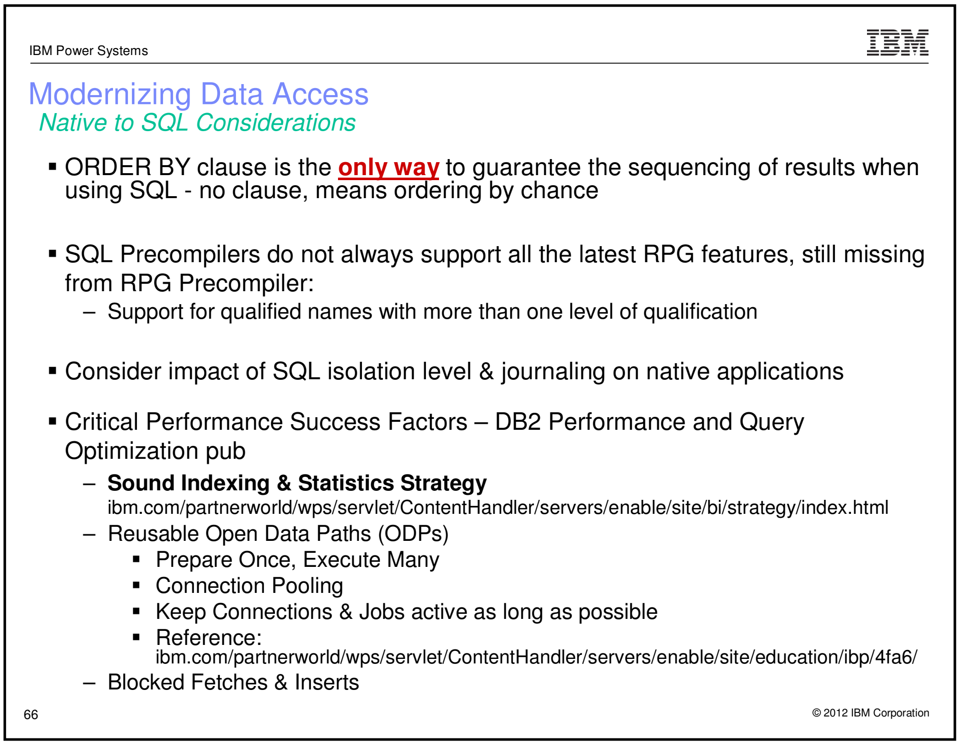
SQL and Scalability

- The issue is throughput not response time
  - As growth occurs, Record Level Access (RLA) will no longer drive POWER based processors
  - Throwing hardware at the problem no longer an option
  - Application changes will be inevitable



Traditional IO does not scale as volumes increase

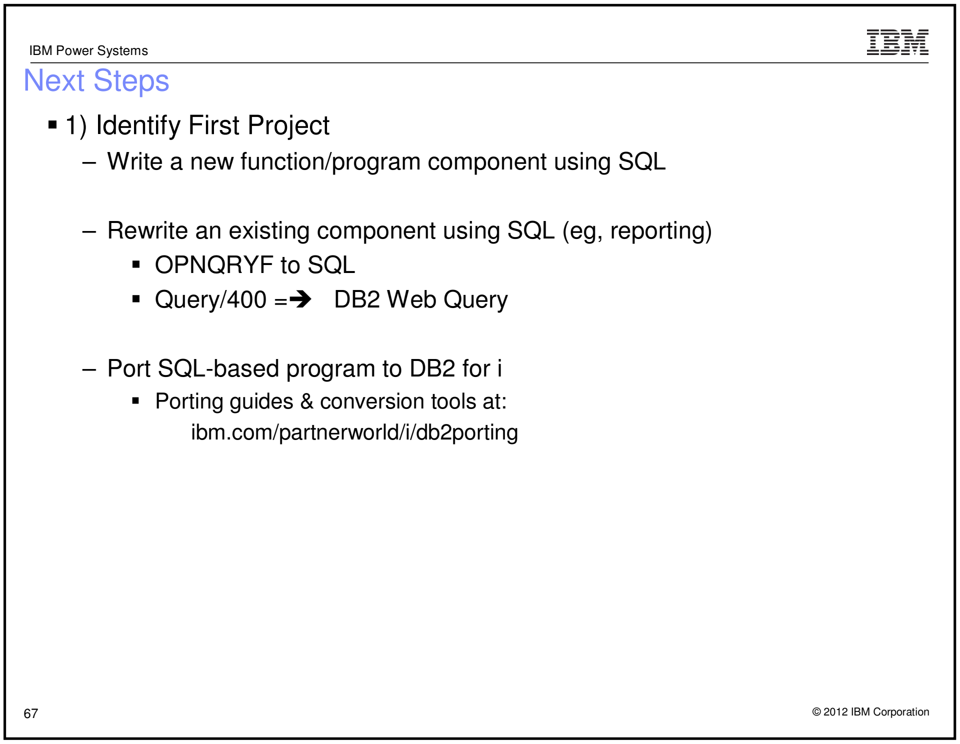
SQL set based access remains flat as growth occurs



Modernizing Data Access

Native to SQL Considerations

- ORDER BY clause is the **only way** to guarantee the sequencing of results when using SQL - no clause, means ordering by chance
- SQL Precompilers do not always support all the latest RPG features, still missing from RPG Precompiler:
  - Support for qualified names with more than one level of qualification
- Consider impact of SQL isolation level & journaling on native applications
- Critical Performance Success Factors – DB2 Performance and Query Optimization pub
  - **Sound Indexing & Statistics Strategy**
    ibm.com/partnerworld/wps/servlet/ContentHandler/servers/enable/site/bi/strategy/index.html
  - Reusable Open Data Paths (ODPs)
    - Prepare Once, Execute Many
    - Connection Pooling
    - Keep Connections & Jobs active as long as possible
    - Reference:
      ibm.com/partnerworld/wps/servlet/ContentHandler/servers/enable/site/education/ibp/4fa6/
  - Blocked Fetches & Inserts



Next Steps

- 1) Identify First Project
  - Write a new function/program component using SQL
  - Rewrite an existing component using SQL (eg, reporting)
    - OPNQRYF to SQL
    - Query/400 =➔ DB2 Web Query
  - Port SQL-based program to DB2 for i
    - Porting guides & conversion tools at:
      ibm.com/partnerworld/i/db2porting

Next Steps

2) Get Education

- *IBM i Database Modernization Workshop*
  http://ibm.com/systems/i/support/itc/educ/lsdb2mod.html

- ***Advanced SQL Workshop –***
  https://ibm.biz/BdDKfg

- *Modernizing iSeries Application Data Access* Redbooks document
  www.redbooks.ibm.com/abstracts/sg246393.html?Open
- *Case Study: Modernizing a DB2 for iSeries Application* white paper
  ibm.com/partnerworld/wps/servlet/ContentHandler/servers/enable/site/education/wp/9e5a/index.html

- DB2 for i SQL Performance Workshop
- ibm.com/systems/i/db2/db2performance.html
  - ibm.com/partnerworld/wps/training/i5os/courses
- Indexing & Stats Strategy White Paper
  ibm.com/partnerworld/wps/servlet/ContentHandler/servers/enable/site/bi/strategy/index.html

- Database modernization roadmaps

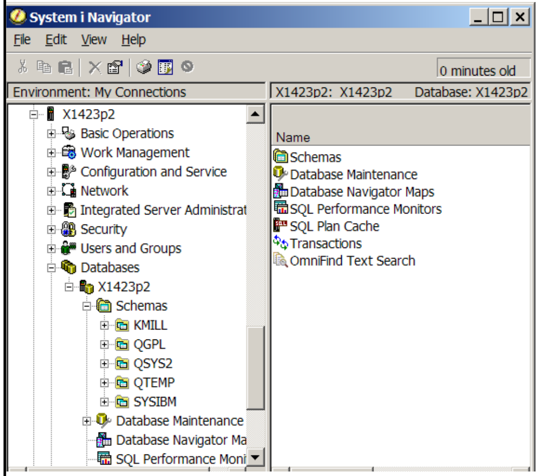
IBM Tooling for DB2 for I and modernization

- IBM DB2 Web Query for i – New Version: 2.1
  - Simplified Packaging
  - New Core-based pricing

- **IBM i Navigator** – DB2 Management w/Visual Explain Run SQL script
- IBM Navigator for i – browser based

- IBM Information Management Products
  - IBM InfoSphere Guardium **V9**
    - Real-time Database Protection & Compliance
    - ibm.com/developerworks/ibmi/library/i-infosphere_guardium_db2/index.html
  - **IBM InfoSphere Data Architect**
  - IBM InfoSphere CDC (Change-Data-Capture)
  - IBM Optim Data Growth Solution
  - IBM Optim Test Data Management & Data Privacy Solution
  - IBM Data Studio
    - SQL and Java Procedure development & debug
    - Wizard-based web service development
    - pureQuery runtime for Java developer productivity

Simplified DB2 for i Management - IBM i Navigator



IBM i 7.1 Enhancements:

OnDemand Performance Center

- Authority Simplification
- Index Advisor Improvements
- Database monitor
  - **Client register filter**
  - Errors only filter
- Show Statements - **Variable replacement**
- Enhanced SQL Details for a Job
  - SQL Monitor integration
  - Connecting QSQSRVR job info

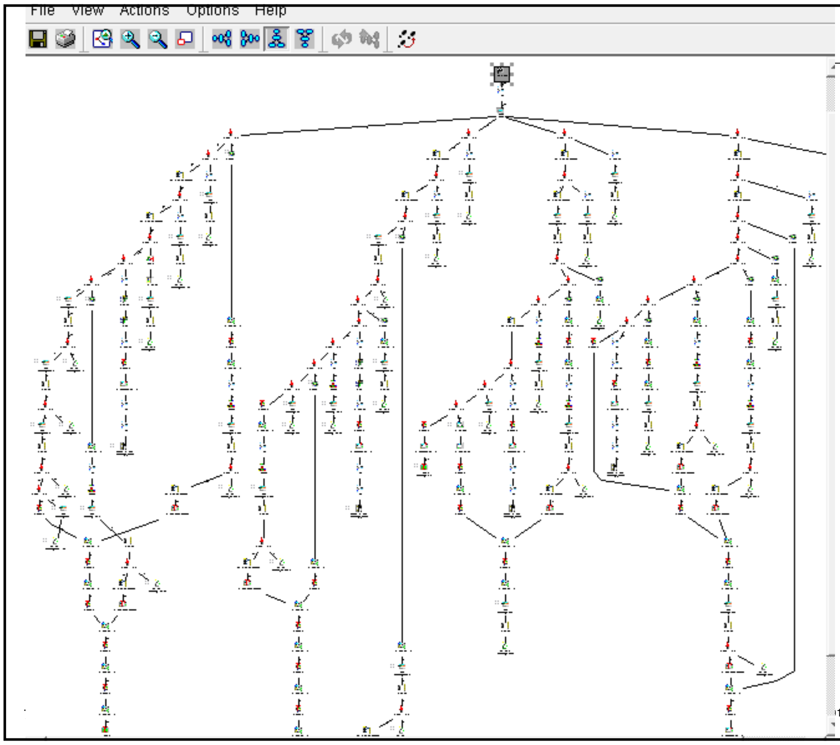
Database Management

- OmniFind Text Index support
- Generate SQL – Privilege & CCSID
- **Progress Status Monitors**
  - Index Build
  - Table Alters
  - Enhanced Reorganize
- Object List enhancements
  - Performance of large lists
  - Object list filtering
  - Save list contents

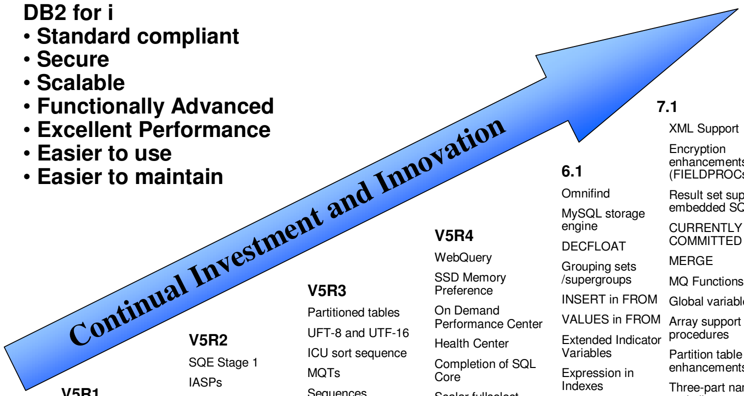
Health Center

- SQL0901 Error Tracker

WHY SQL?



DB2 for i
- Standard compliant
- Secure
- Scalable
- Functionally Advanced
- Excellent Performance
- Easier to use
- Easier to maintain



V5R1

SQL triggers
Java Functions
DRDA DUW TCP/IP
2 GB LOBs
1 Terabyte Table
Journal Minimal Data
Two-phase over TCP/IP
DDL Journaling
Database Navigator
Generate SQL

V5R2

SQE Stage 1
IASPs
Identity columns
Savepoints
UNION in views
Scalar subselect
UDTFs
DECLARE GLOBAL TEMPORARY TABLE
Catalog views
JDBC V3.0
DRDA Kerberos
Journal Standby

V5R3

Partitioned tables
UFT-8 and UTF-16
ICU sort sequence
MQTs
Sequences
Implicit char/numeric
BINARY/VARBINARY
GET DIAGNOSTICS
DRDA Alias
DECIMAL(63)
SQE Stage 3
Ragged SWA
QDBRPLAY
Online Reorganize

V5R4

WebQuery
SSD Memory Preference
On Demand Performance Center
Health Center
Completion of SQL Core
Scalar fullselect
Recursive CTE
INSTEAD OF triggers
Descriptor area
XA over DRDA
DDM 2-phase
Scrollable cursor
2M SQL statement
1000 tables in a query
SQE Stage 5
Implicit journaling enhancements

6.1

Omnifind
MySQL storage engine
DECFLOAT
Grouping sets /supergroups
INSERT in FROM
VALUES in FROM
Extended Indicator Variables
Expression in Indexes
ROW CHANGE TIMESTAMP
Statistics catalog views
CLIENT special registers
SQE Stage 6
DDM and DRDA IPv6
Deferred Restore of MQT and Logicals
Environmental limits

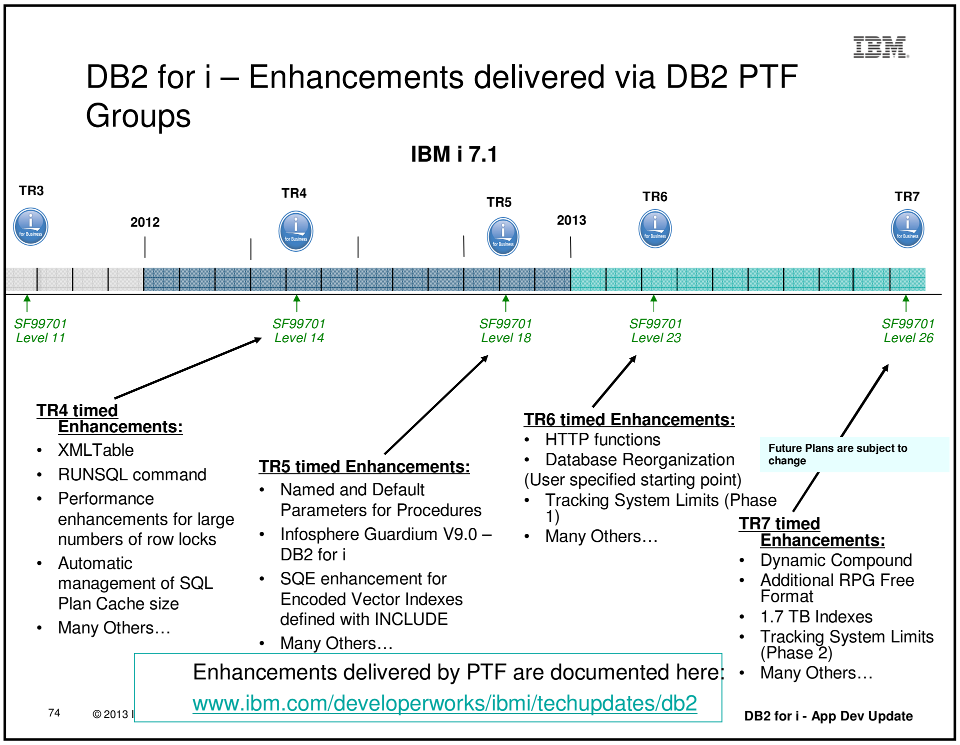
7.1

XML Support
Encryption enhancements (FIELDPROCs)
Result set support in embedded SQL
CURRENTLY COMMITTED
MERGE
MQ Functions
Global variables
Array support in procedures
Partition table enhancements
Three-part names and aliases
SQE Logical file support
SQE Adaptive Query Processing
EVI enhancements
Inline functions
CREATE OR REPLACE

Next

Major enhancement
XMLTABLE
CONNECT BY
OLAP Extensions
Regression Functions/Covariance/Correlation
TRANSFER OWNERSHIP
Named arguments and defaults for parameters
Obfuscation of SQL routines
Array support in UDFs
Timestamp precision
Multiple-action Triggers
Built-in Global Variables
Record movement between partitions on UPDATE
1.7 Terabyte Indexes
Health Center – Non-database limits
Navigator Graphing and Charting

IBM Information Management software



DB2 for i – Enhancements delivered via DB2 PTF Groups

IBM i 7.1

TR4 timed Enhancements:

- XMLTable
- RUNSQL command
- Performance enhancements for large numbers of row locks
- Automatic management of SQL Plan Cache size
- Many Others…

TR5 timed Enhancements:

- Named and Default Parameters for Procedures
- Infosphere Guardium V9.0 – DB2 for i
- SQE enhancement for Encoded Vector Indexes defined with INCLUDE
- Many Others…

TR6 timed Enhancements:

- HTTP functions
- Database Reorganization

(User specified starting point)

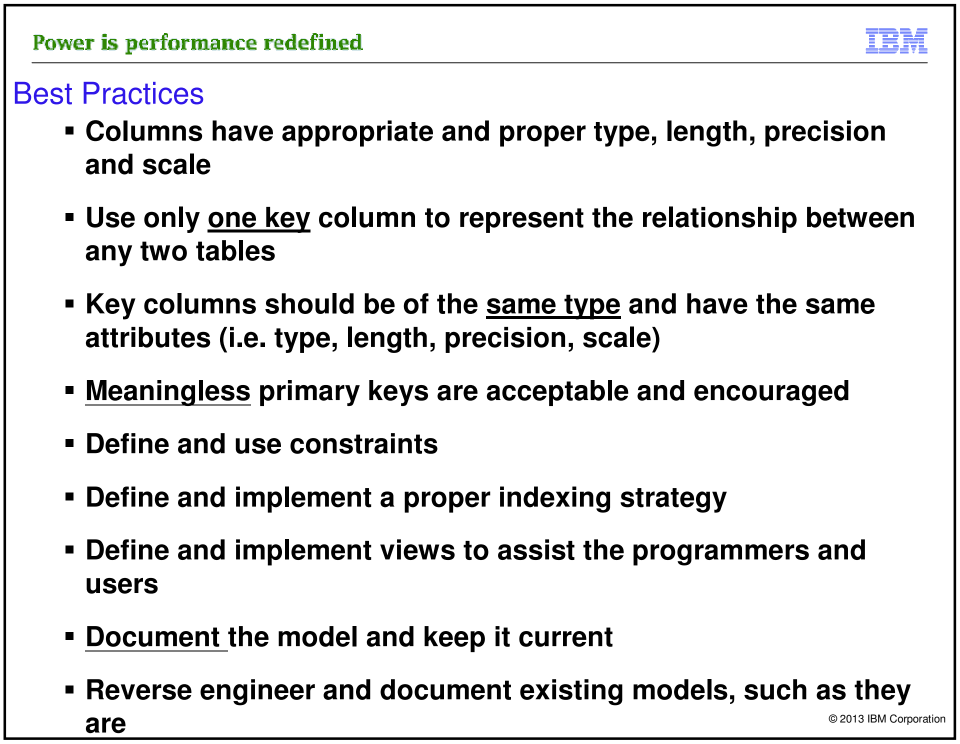
- Tracking System Limits (Phase 1)
- Many Others…

Future Plans are subject to change

TR7 timed Enhancements:

- Dynamic Compound
- Additional RPG Free Format
- 1.7 TB Indexes
- Tracking System Limits (Phase 2)
- Many Others…

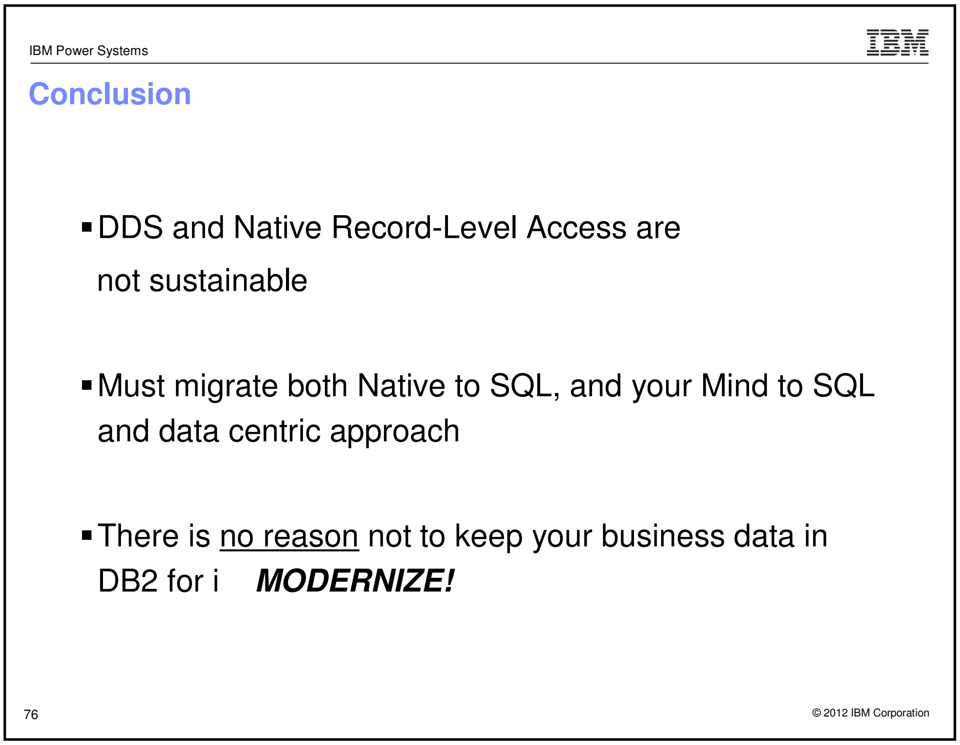
Enhancements delivered by PTF are documented here:
www.ibm.com/developerworks/ibmi/techupdates/db2



Power is performance redefined

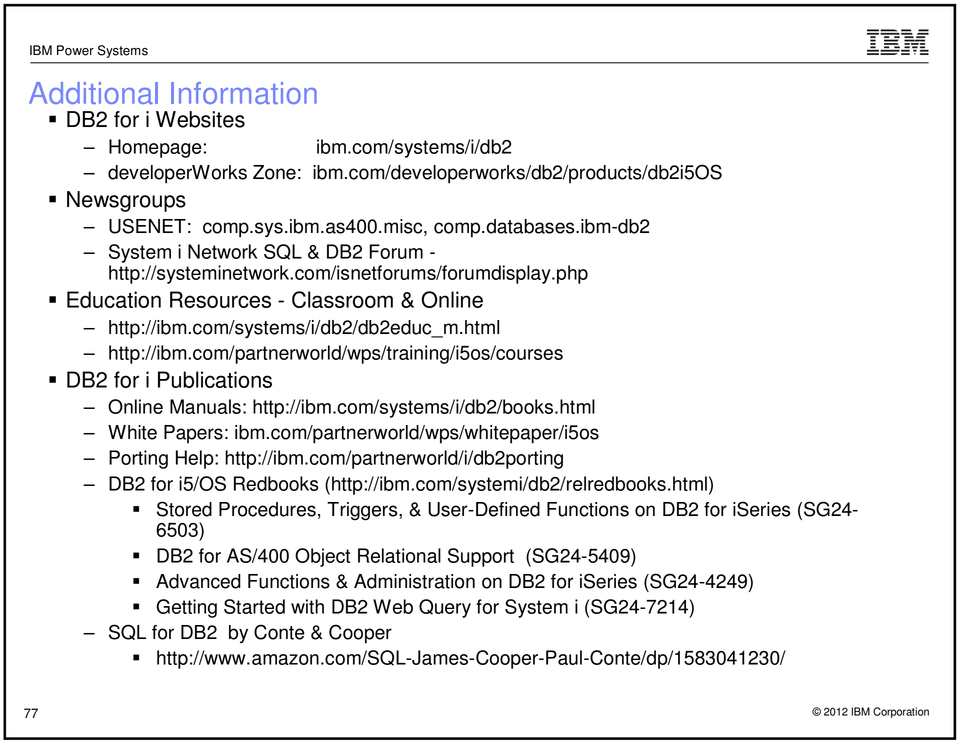
Best Practices

- Columns have appropriate and proper type, length, precision and scale
- Use only one key column to represent the relationship between any two tables
- Key columns should be of the same type and have the same attributes (i.e. type, length, precision, scale)
- Meaningless primary keys are acceptable and encouraged
- Define and use constraints
- Define and implement a proper indexing strategy
- Define and implement views to assist the programmers and users
- Document the model and keep it current
- Reverse engineer and document existing models, such as they are



Conclusion

- DDS and Native Record-Level Access are not sustainable
- Must migrate both Native to SQL, and your Mind to SQL and data centric approach
- There is no reason not to keep your business data in DB2 for i ***MODERNIZE!***



Additional Information

- DB2 for i Websites
  - Homepage: ibm.com/systems/i/db2
  - developerWorks Zone: ibm.com/developerworks/db2/products/db2i5OS
- Newsgroups
  - USENET: comp.sys.ibm.as400.misc, comp.databases.ibm-db2
  - System i Network SQL & DB2 Forum - http://systeminetwork.com/isnetforums/forumdisplay.php
- Education Resources - Classroom & Online
  - http://ibm.com/systems/i/db2/db2educ_m.html
  - http://ibm.com/partnerworld/wps/training/i5os/courses
- DB2 for i Publications
  - Online Manuals: http://ibm.com/systems/i/db2/books.html
  - White Papers: ibm.com/partnerworld/wps/whitepaper/i5os
  - Porting Help: http://ibm.com/partnerworld/i/db2porting
  - DB2 for i5/OS Redbooks (http://ibm.com/systemi/db2/relredbooks.html)
    - Stored Procedures, Triggers, & User-Defined Functions on DB2 for iSeries (SG24-6503)
    - DB2 for AS/400 Object Relational Support (SG24-5409)
    - Advanced Functions & Administration on DB2 for iSeries (SG24-4249)
    - Getting Started with DB2 Web Query for System i (SG24-7214)
  - SQL for DB2 by Conte & Cooper
    - http://www.amazon.com/SQL-James-Cooper-Paul-Conte/dp/1583041230/

DB2 Modernization Assistance

DB2 for i Modernization Workshop
http://ibm.com/systems/i/support/itc/educ/lsdb2mod.html

→Need help?

IBM DB2 for i Consulting and Services

- ✓ Database modernization
- ✓ DB2 Web Query
- ✓ Database design, features and functions
- ✓ DB2 SQL performance analysis and tuning
- ✓ Data warehousing and Business Intelligence
- ✓ DB2 for i education and training

Contact: Mike Cain mcain@us.ibm.com
IBM Systems and Technology Group
Rochester, MN USA