Architektur und Betrieb von kommerziellen Anwendungssystemen

Power Systems

Inhalt: Rechnerarchitektur, Softwarearchitektur, Systemadministration und -betrieb müssen aufeinander abgestimmt sein, um für ein kommerziell eingesetztes Informationssystem eine hohe Leistung bei gleichzeitig geringen Betriebskosten zu erzielen. Typische Probleme sind dabei Antwortzeitverhalten, Durchsatz, Sicherheit, Schutz vor Datenverlust, Serverkonsolidierung, Skalierbarkeit, Hochverfügbarkeit und die Integration existierender Infrastruktur. In dieser Lehrveranstaltung werden mögliche Lösungen und die Vorteile einer integrierten Betriebssystemumgebung am Beispiel einer iSeries Umgebung ganzheitlich studiert. In den begleitenden Übungen arbeiten die Teilnehmer an einem IBM eServer i5.

Dauer: 2h Vorlesung, (optional 2h Lab)

Einordnung: SP 2 (Rechnerarchitektur, eingebettete Systeme und Simulation)
SP 5 (Sicherheit und Verifikation)

Lab (optional): Die Übungen können über eine VPN-Verbindung und mittels Remote Desktop an einem 4-Prozessor eServer i5 Modell 550 durchgeführt werden.

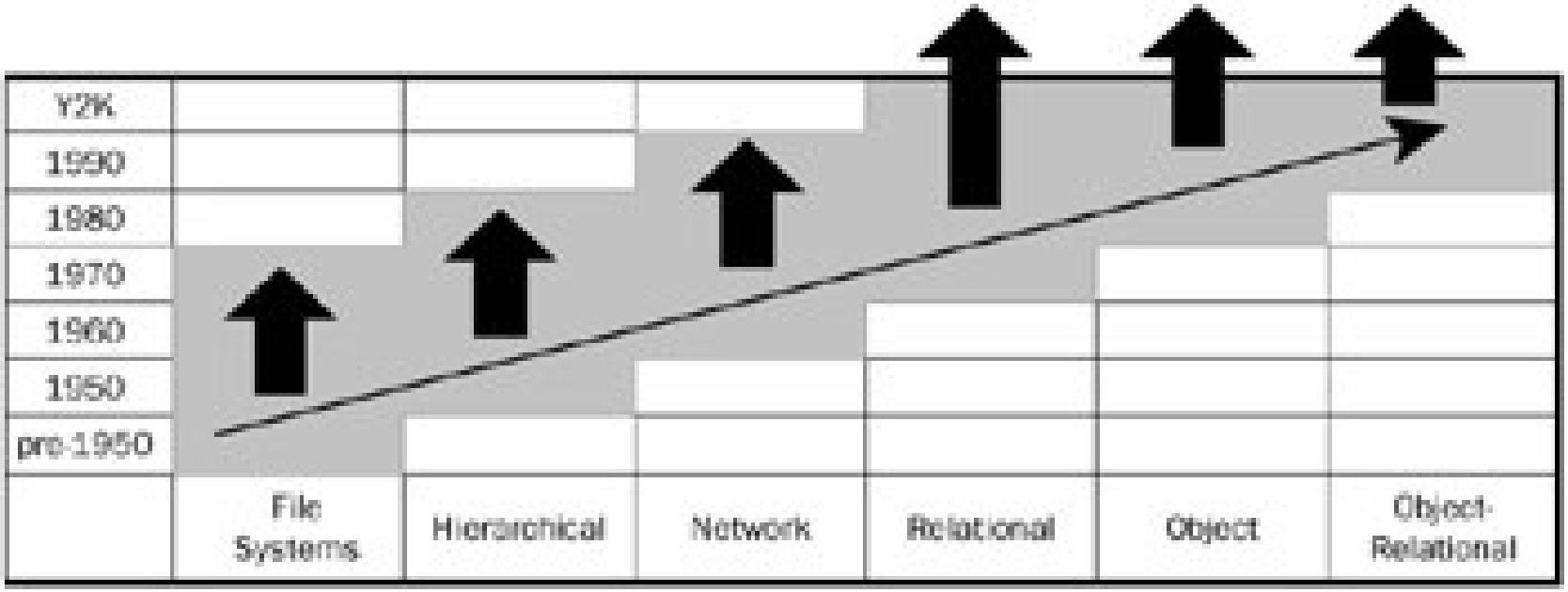
Content (planned)

Power Systems

- Requirements for Modern Application Systems
- i5/OS Architecture
- OS Security
- Work Management
- Networking
- **Databases and Filesystems**
- Consolidation of Windows Environments
- Storage Consolidation
- e-business Infrastructure
- High Availablility
- Backup and Restore
- Logical Partioning
- Virtualization Technologies

8.1 DB Basics

The Evolution of Database Modeling Techniques



Source: “Beginning Database Design and Implementation”, Gavin Powell, ISBN:0764574906

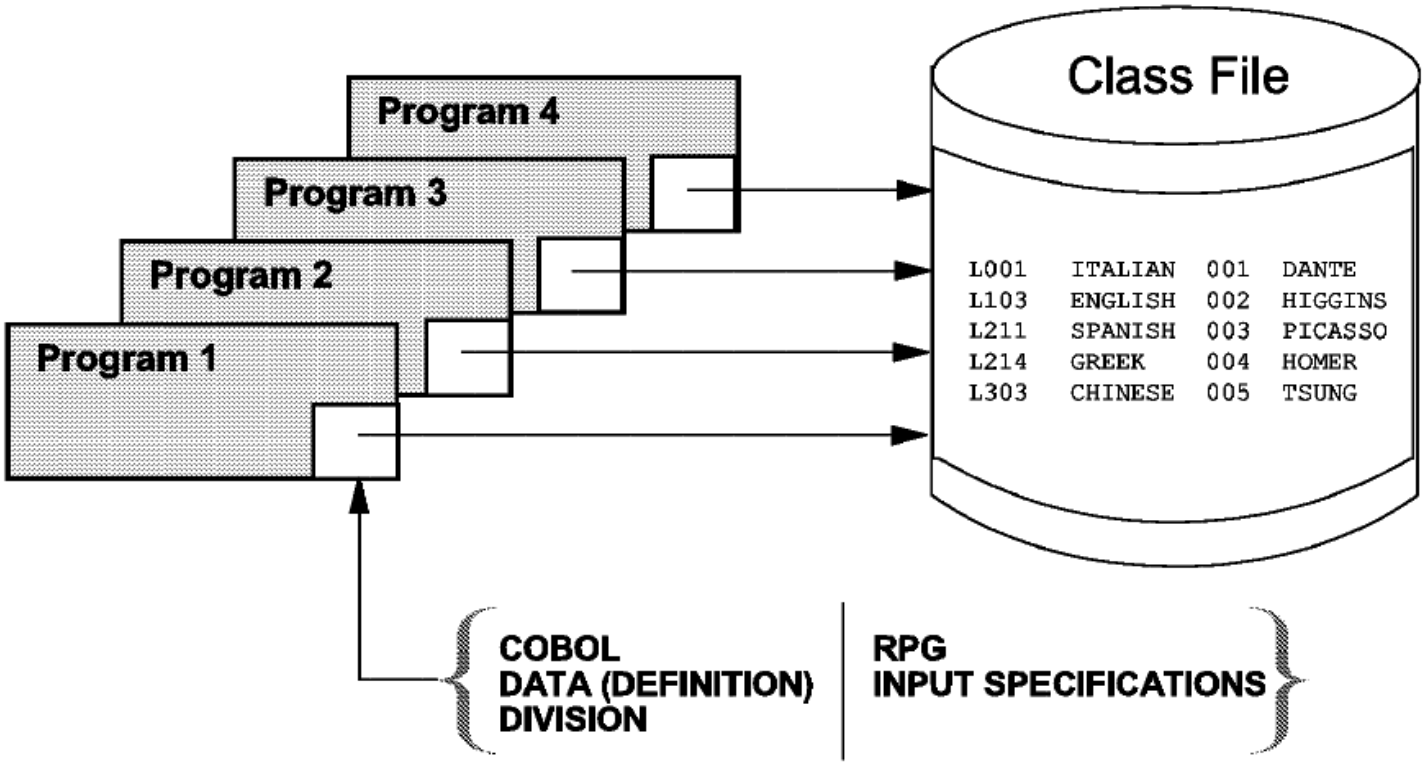
File Systems

- data are stored in flat files
  - no data types
  - no indexing, any searching through flat files for data has to be explicitly programmed
- data or better file management have to be done by operating system
- CSV files are still very popular

Program-Described Database File

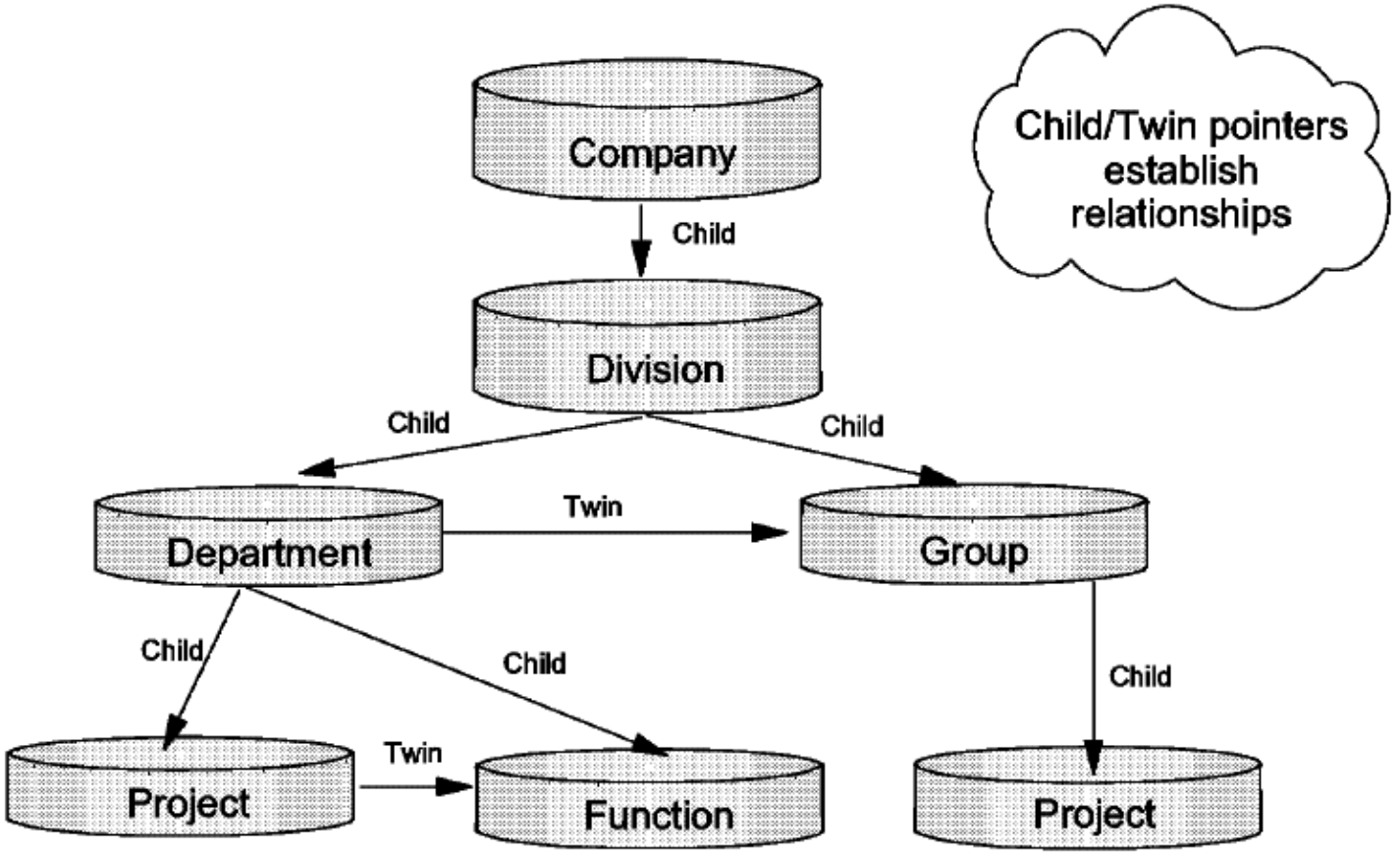
Power Systems

Each Program has it‘s own data definitions!



Changing a file's format requires changing data definition in each program

Hierarchical Database



Information Management System (IMS)

- hierarchical database
- IBM introduced IMS in 1968
  - simple structure
  - performance
    - IMS can store parent/child close to another minimizing disk I/O
- still very widely used on IBM mainframes
  - examples: ATM transactions

Source: “SQL: The Complete Reference”, James R.Groff, Paul N.Weinberg, McGraw-Hill/Osborne 2002

Network Database

- extends hierarchical database structure
  - allows childs to have more than one parent
- 1971 standardization (named CODASYL)

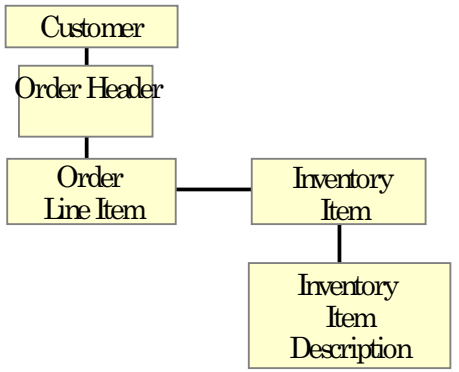
Disadvantages of hierachical and network databases:

- Changing database structure typically required rebuilding the entire database
- Programs have to be written for every report

Relational Database Architecture

Power Systems

- Created to remove program maintenance of data
  - Invented by IBM, E.F. Codd's 12 rules
- Data Integrity, Relationships, Security, Business Rules
  - Managed by RDBMS, not applications
- Isolate Programmer from managing data on physical disk units
  - DBA's created to now manage at the database level
    - (on most client/server RDBMS's)
- Concepts of RDBMS
  - Tables, Columns, Views, Indexes
    - Physical Files, Logical Files, Members
  - Referential Integrity
  - Triggers, Stored Procedures
  - Concurrency
- SQL (Structured Query Language)
  - The "open" RDBMS programming language
  - JDBC, ODBC, CLI, SQL, Query Tools



STRUCTURED QUERY LANGUAGE (SQL)

Select A.CUSTOMER, B.ORDER_TOTAL, C.PRODUCT
From Customer_Master A, Order_Header B, Order_Line_Item C,
Where A.CUSTID =B.CUSTID AND B.ORDER_ID=C.ORDERID AND A.REGION="Central"

The Business Problem

- The Relational Database Management Systems were not an effective manager of unstructured or semi-structured data (documents, images, e-mails, engineering drawings, video clips, and other business formats):
  - Most of these data types remain outside the database in a file system
  - These files are often related to traditional data stored in the database
- Users want a content management system that integrates:
  - Management of the file and its associated data
  - Synchronizes updates, backup, recovery, and other functions across both the RDBMS and the file system

Object-Oriented DBMS

- in the 90th OO became popular
- research on OO databases
- no standardized query language like SQL
- not sucessful in the marketplace

Object-Relational Databases

- extends relational database model
- some object-oriented capabilities are added

Unstructered Datas

Power Systems

- Complex Objects
  - LOBs
    - Large Objects stored in table columns

Name	Address	Picture
John Doe	123 First Ave	

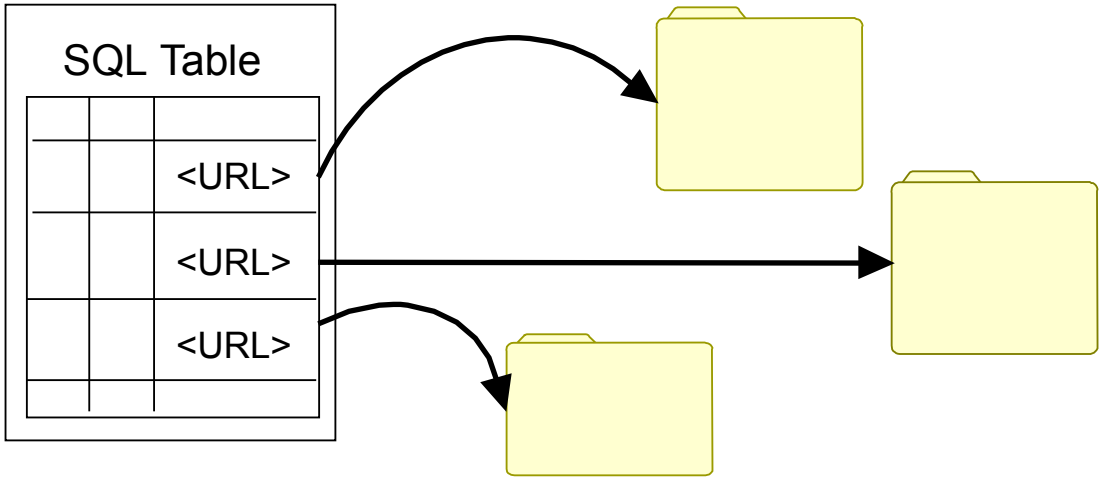
  - DataLinks
    - Downloadable to PC Applications

Name	Address	Picture
John Doe	123 First Ave	URL

File System

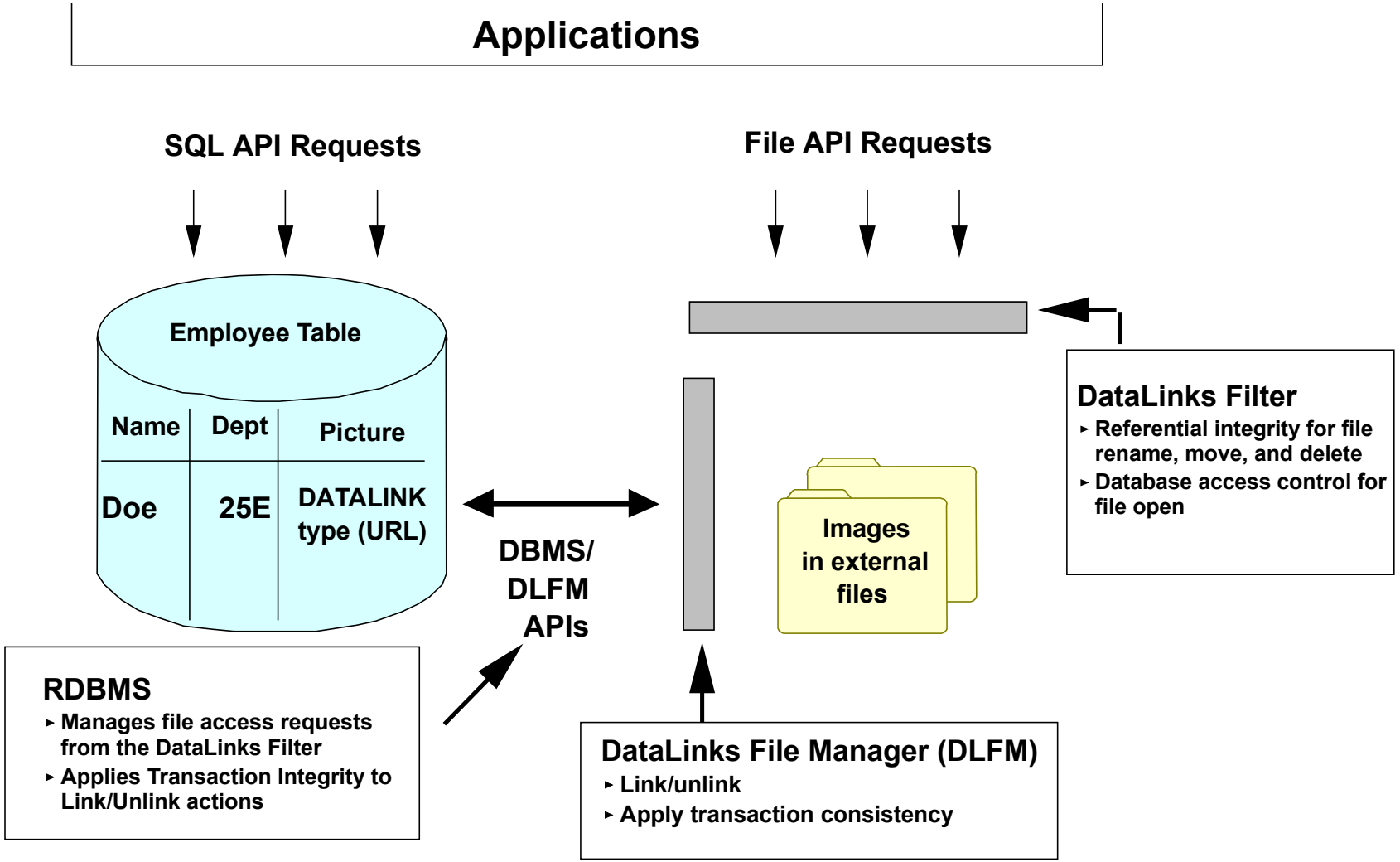
Datalink

- **The DataLink data type is one of the basic building blocks for extending the types of data that can be stored in database files**
  - The DATALINK column holds information about an object, without actually containing the object itself
  - The data stored in the column is a pointer to the object
    - A Uniform Resource Locator (URL) is stored
  - The object can be anything (image file, voice recording, text file, and so forth)
  - The URL is used to resolve to the object
  - There is a relationship between the row and the object.



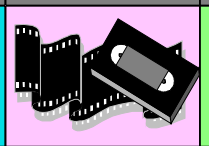
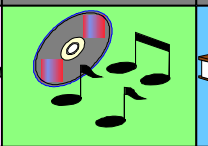
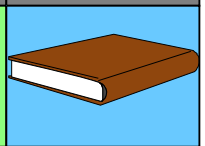
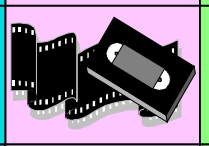
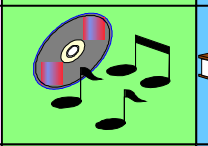
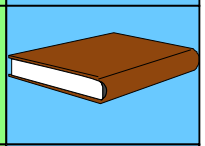
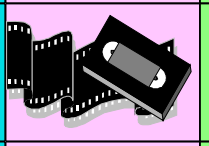
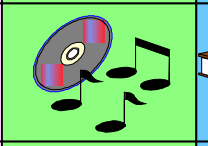
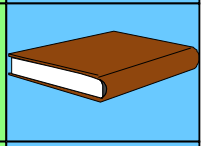
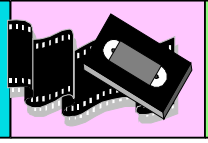
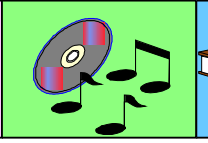
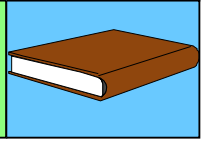
DataLinks Components

Power Systems



The Need for Large Objects (LOBs)

Power Systems

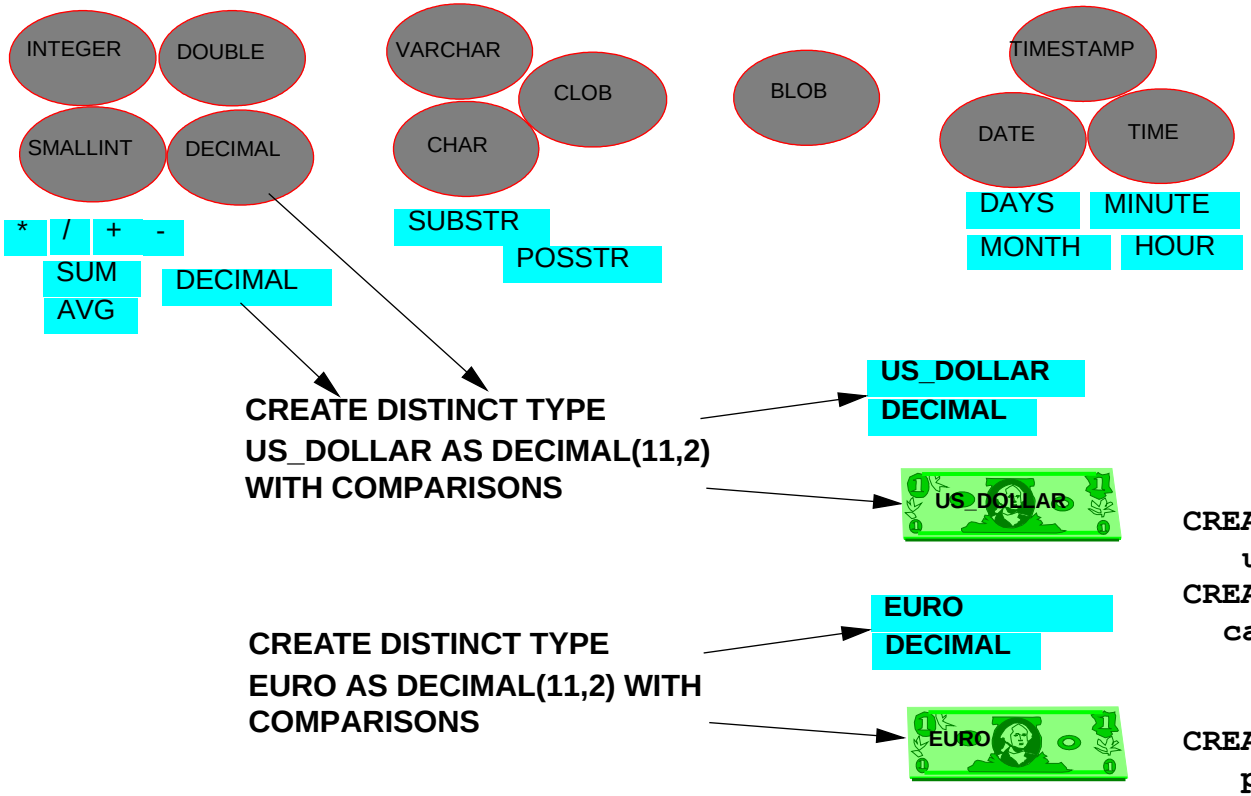
SOLD	ONHAND	RATING	ARTIST	TITLE	COVER	VIDEO	MUSIC	SCRIPT
234	59	PG-13	Arnold	The Exterminator				
13	45	R	Kevin	Dancing with Bulls				
1295	209	G	Glenn	101 Doll Imitations				
379	112	G	Buzz	Toy Glory				

- Provide support for Web-based multimedia applications
- Provide a built-in data type that can store a large amount of data

- There are three different types of data:
  - BLOB (Binary Large Object): For scanned documents, digital images, and so forth
  - CLOB (Character Large Object): Large character string data type
  - DBCLOB (Double Byte Character Large Object): Large double-byte character string data type

User-Defined Types - Example

Power Systems




```
CREATE DISTINCT TYPE
   us_dollars AS DECIMAL(11,2);
CREATE DISTINCT TYPE
  can_dollars AS DECIMAL(11,2);


CREATE TABLE products (
   prod_id       CHAR(5),
   prod_desc     VARCHAR(50),
   prod_weight   DECIMAL(11,2),
   usa_price     US_DOLLARS,
   can_price     CAN_DOLLARS,
   product_imageBLOB(1M)
   )
```

User-Defined Functions - Examples

Power Systems

FIND_PICTURE(,SLIDE_SHOW)

Large Object Search

15,
78, 54, 67, 67, 94,
99, 53, 47, 87, 72,
34, 63, 54, 59, 82

ARRAY_AVERAGE(CLASSROOM_MARKS)

Built-in Functions:

SQRT() EXP() SIN() COS()
TAN() LOG() RAND() POWER()

Type Cast Functions:

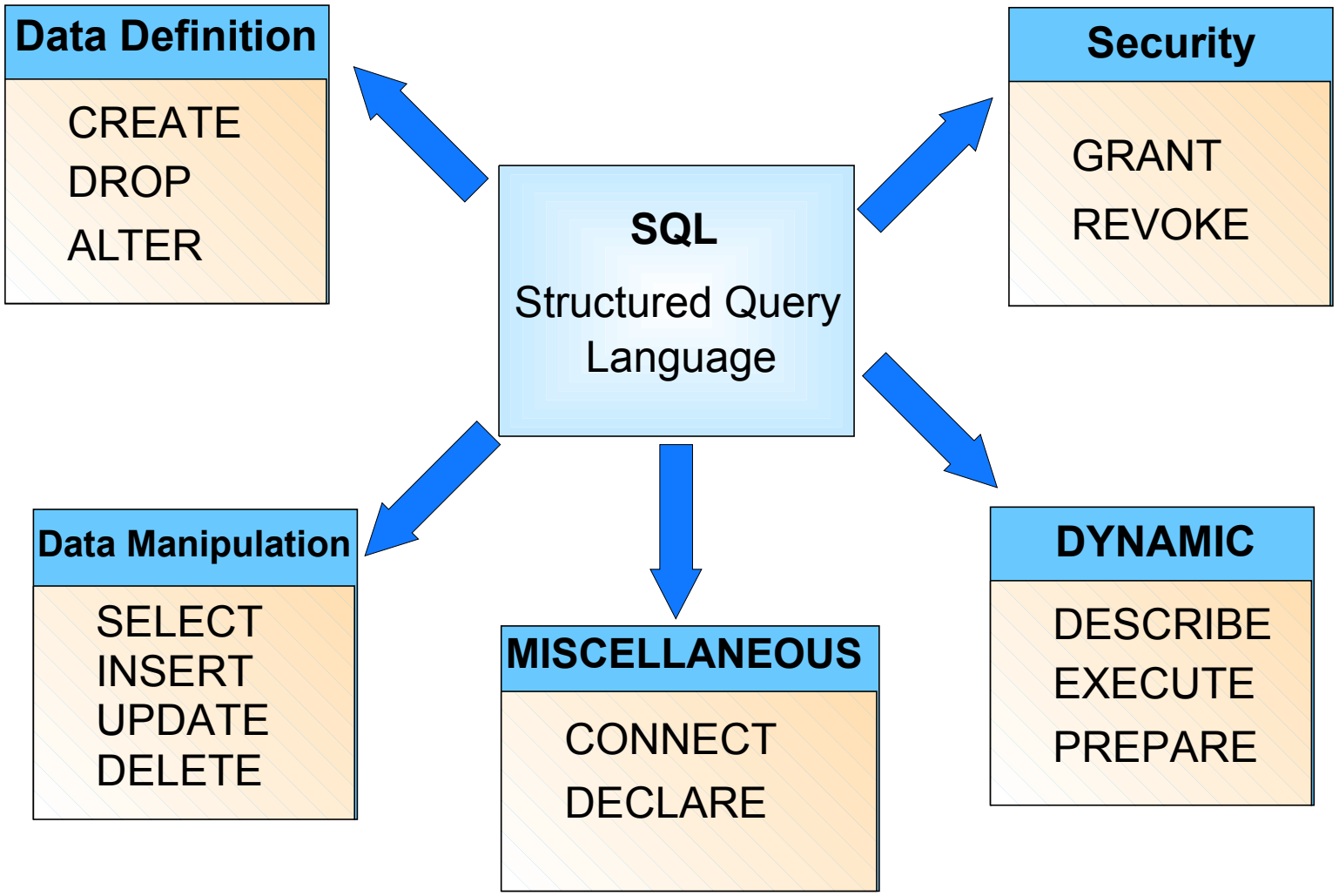
MONEY() DOLLAR()

Structured Query Language

- Portability of code & skills
- Strategic database interface for industry & i5/OS
  - SQL required for certain functions & middleware
- J2EE architecture based on SQL interfaces
  - Data types: BLOB, CLOB, Datalink, ...
  - Auto-Incrementing Constructs: Sequence & Identity column attribute
  - Column-level Triggers
  - Encryption & Decryption functions
  - Encoded Vector Indices
  - InsteadOf Trigger
  - ...
- SQL as a programming language can reduce total lines of code
- DB2 Symmetrical Multiprocessing - parallel database processing

SQL Components

Power Systems



DBMS Architectures

- Enterprise DBMS
  - Designed for scalability and high performance,
    - very large databases
    - large number of concurrent user
    - multiple types of application
- Departmental DBMS
  - For small or medium-sized workgroups
- Personal DBMS
  - Single user environment
- Mobile DBMS
  - Specialized form of a departmental or enterprise DBMS
  - Needs synchronization for remote database changes

Therefore, every vendor have different products.

Types of Databases

- Transactional Databases
  - based on small changes to the database
  - transaction oriented
- Decision Support Databases
  - Data warehouse database
  - Data mart
  - Reporting database
- Hybrid Databases
  - a mixture of both

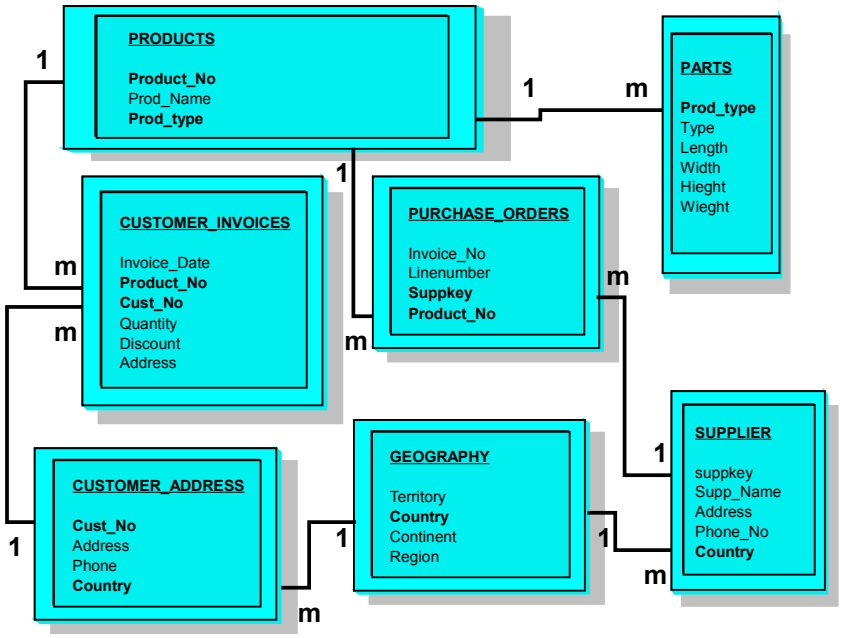
OLTP - Online Transaction Processing

Power Systems

- **Online Transaction Processing is a "data handling method"**

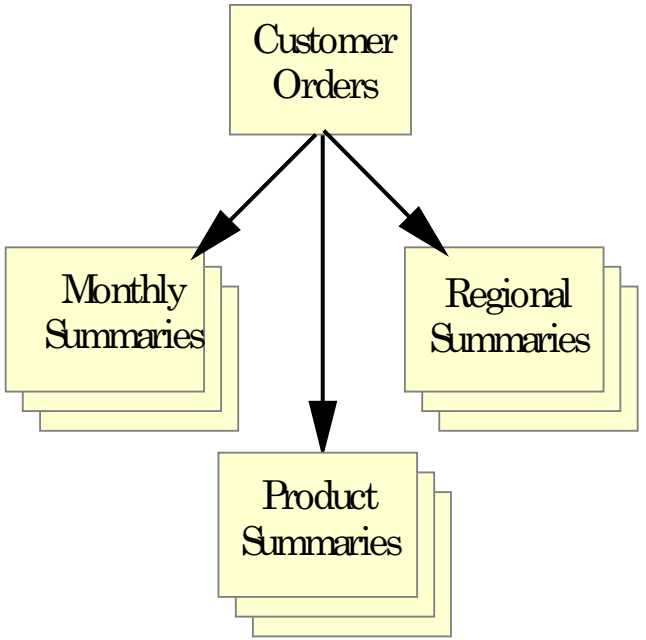
- **Characteristics :**
  - **detail oriented**
  - **highly repetitive**
  - **end-users next action is predictable to a certain degree**

- **Examples**
  - **order entry**
  - **payroll applications**
  - **inventory management applications**
  - **... all applications that support "day-to-day" business requirements**

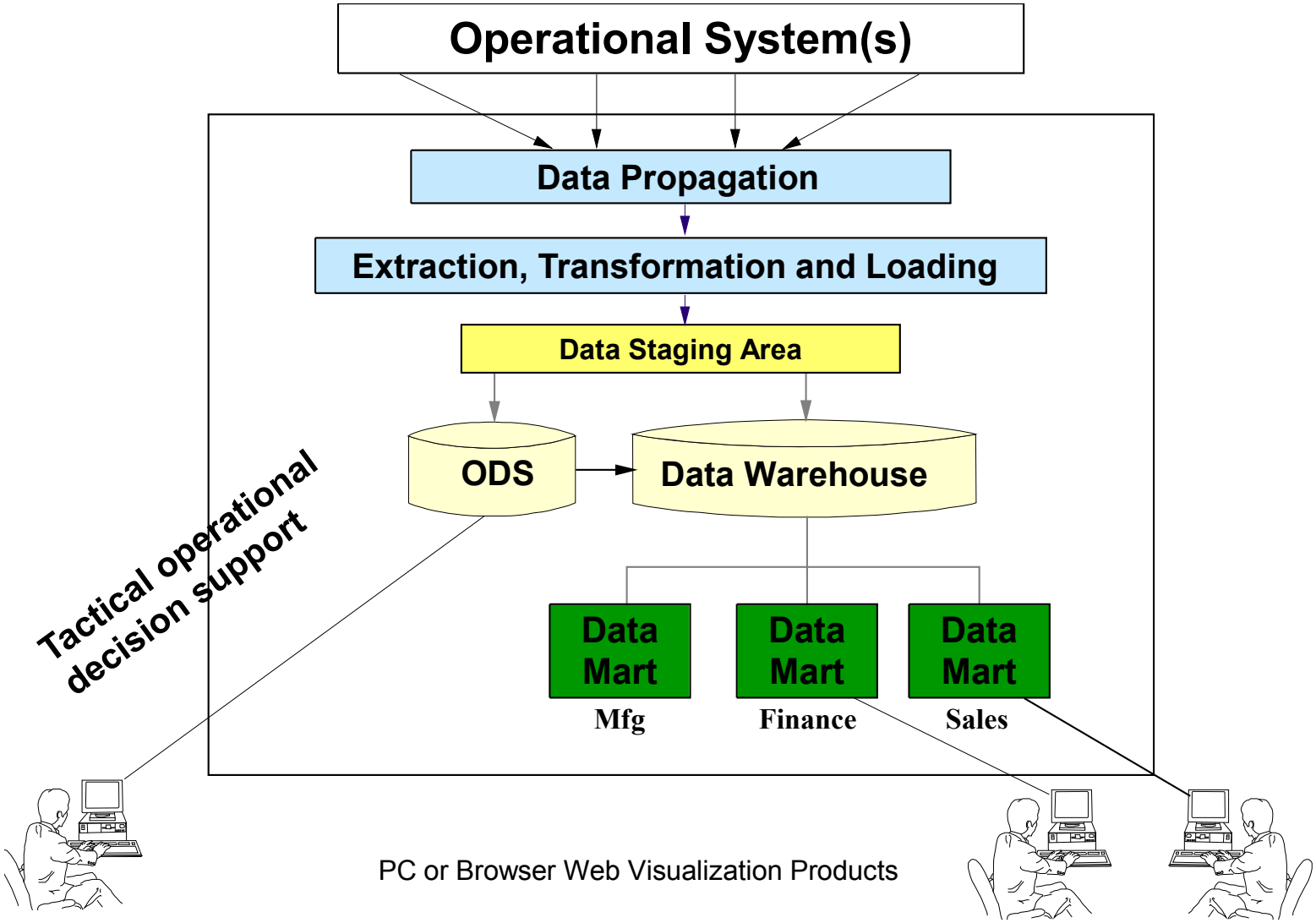


OLAP - Online Analytical Processing

- **OLAP databases are typically:**
  - **Replicated summarized (de-normalized / highly indexed) and cleansed data created from operational transaction systems**
  - **Placed on separate database server**
  - **Refreshed on timed basis from operational transactions**
  - **Multidimensional in nature - read only databases**
  - **Could be quite LARGE (several years of historical data)**



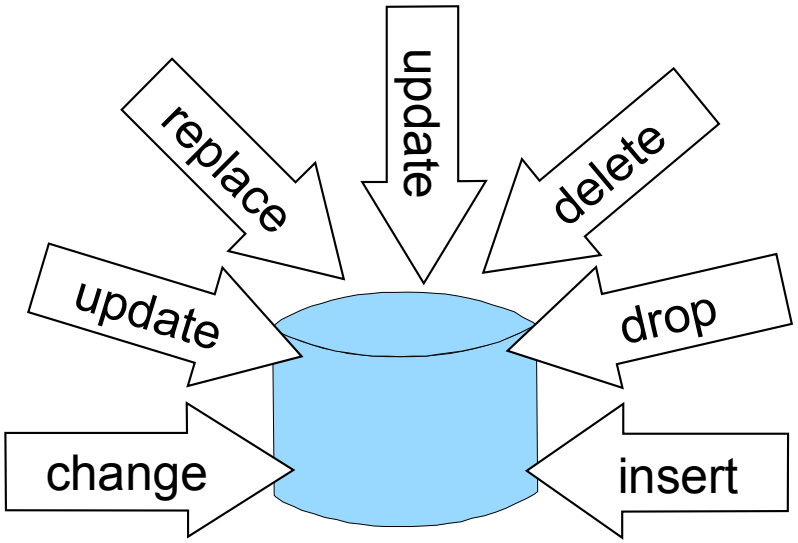
The Enterprise DW Architecture



OLTP versus OLAP Databases

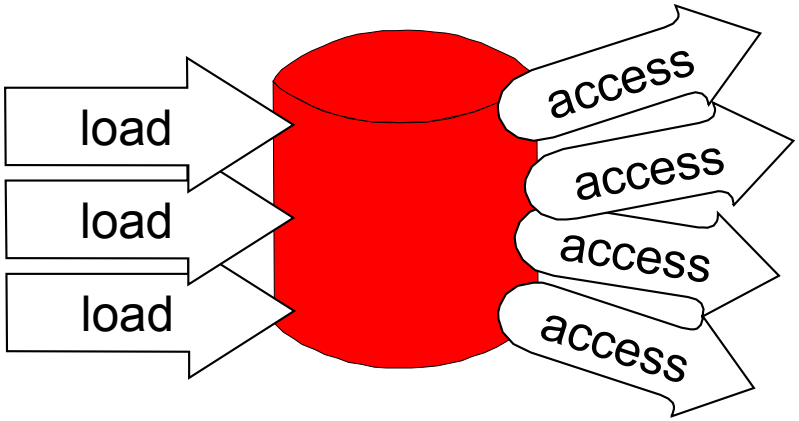
Power Systems

OLTP



data is regularly updated on a record by record basis

OLAP



data is loaded into the data warehouse and is accessed there but is not updated

8.2 System i Integrated Relational Database Management System

DB2 UDB for IBM i

IBM i and DB2 UDB Family

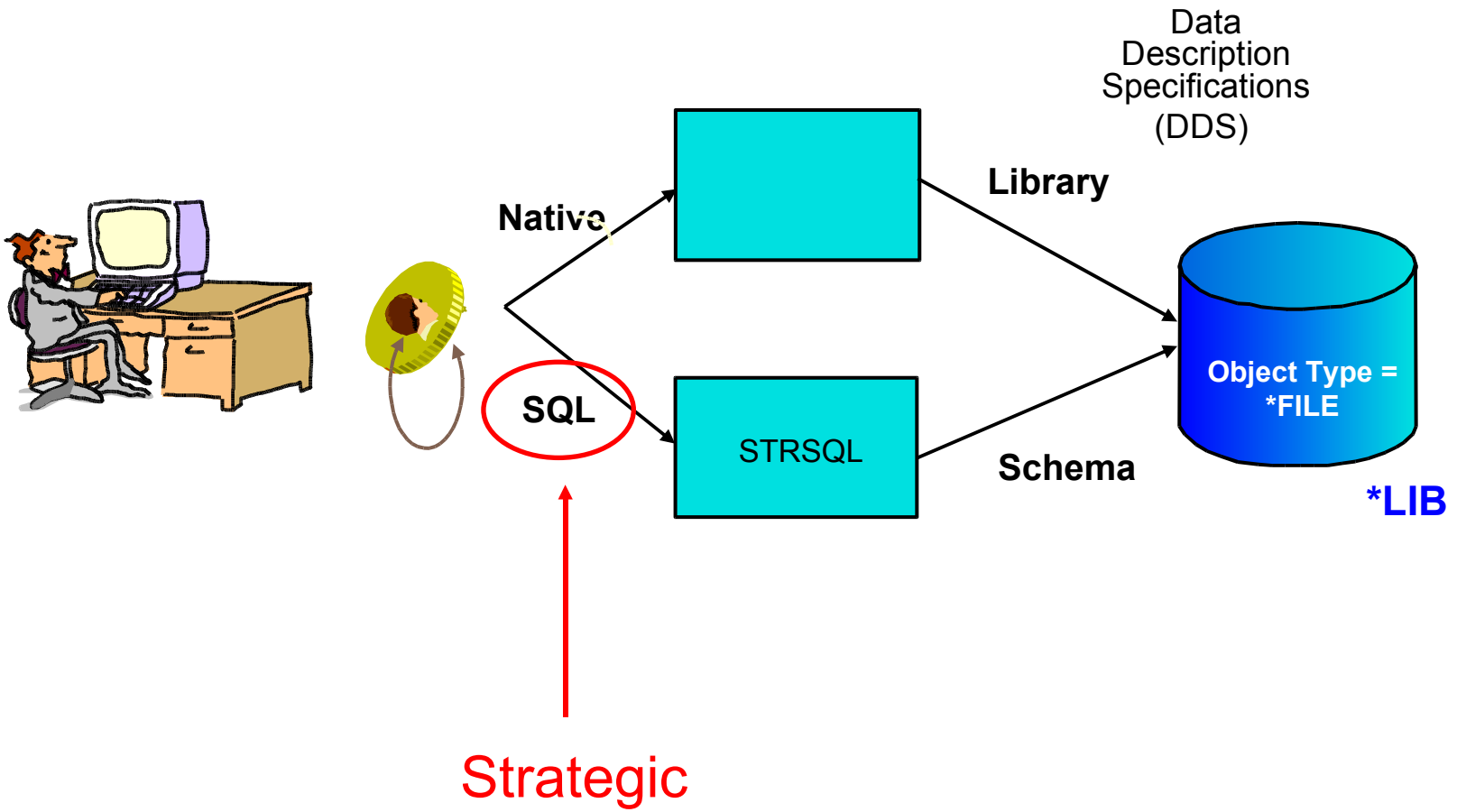
Power Systems

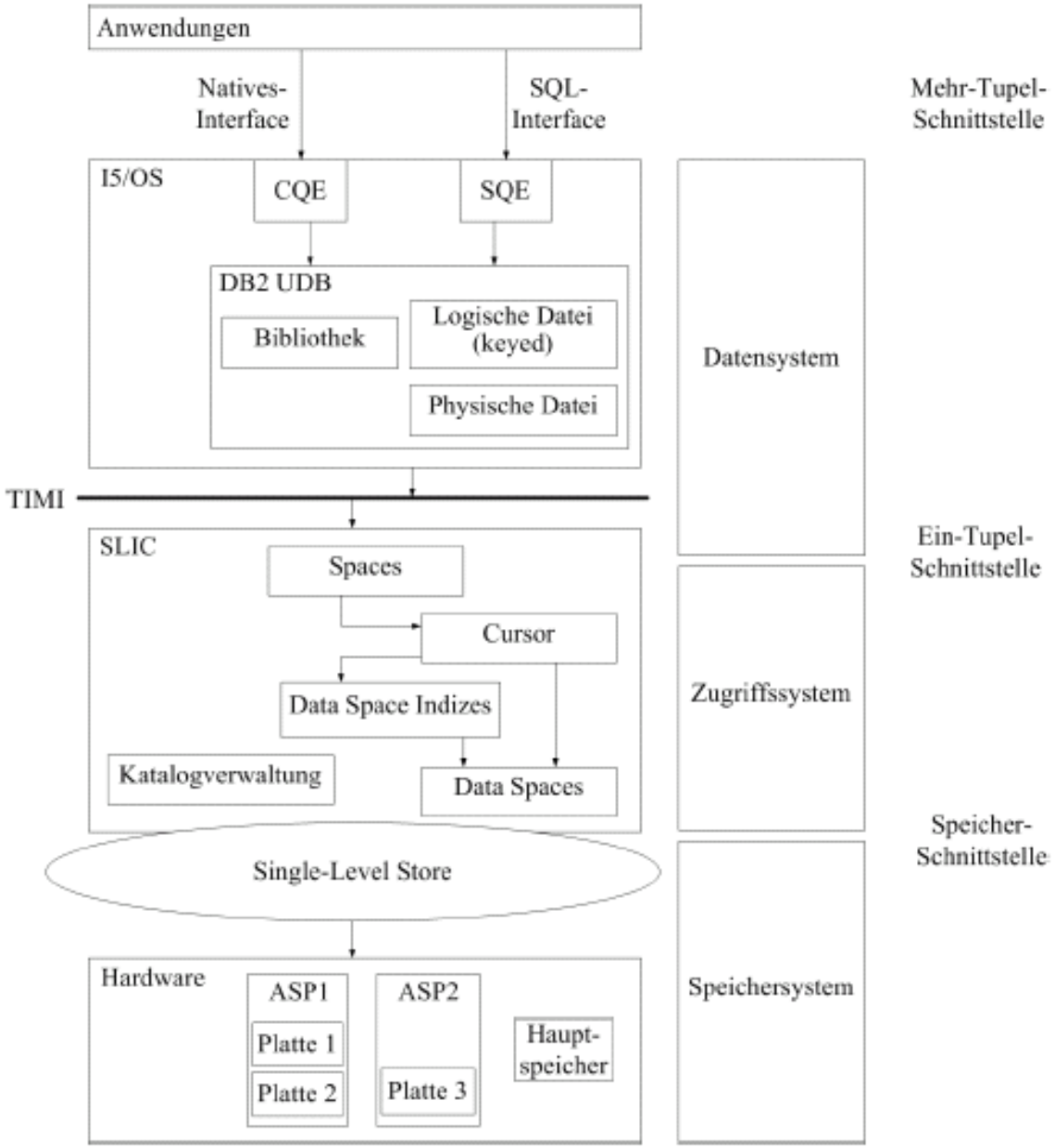
- Similarities
  - SQL Syntax
  - Universal Functionality
  - Open, Common Interfaces
    - DRDA, ODBC, JDBC, CLI
  - Common Tools and Middleware

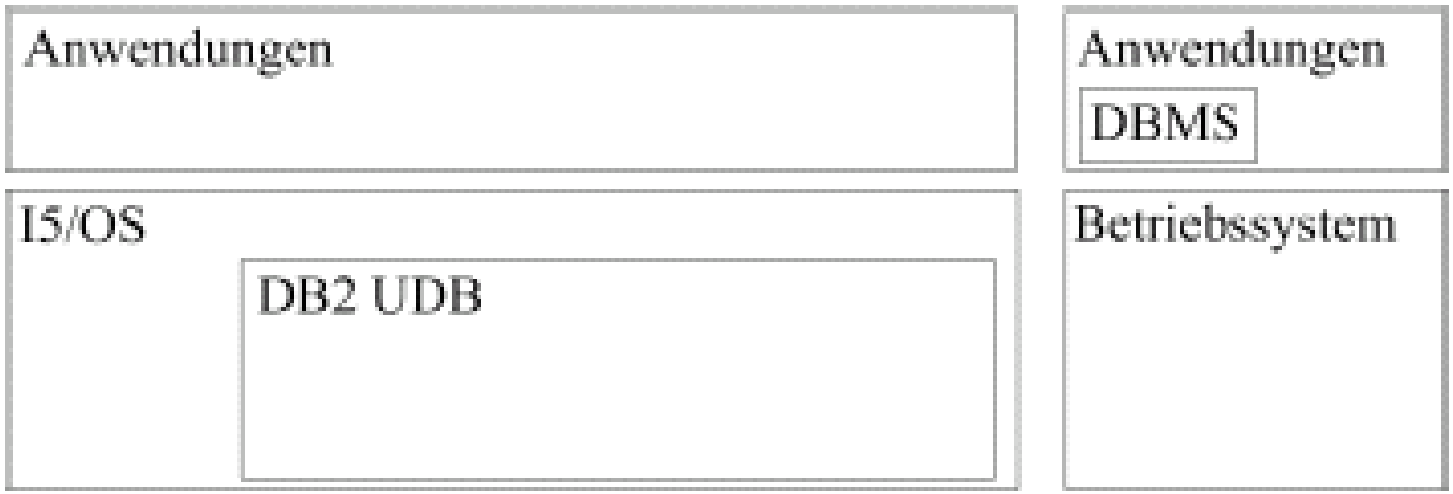
- Differences
  - Not the same codebase
    - Some algorithms and technology is ported
  - Administration and Integration
    - i5/OS have a single system-wide database instance, no need for multiple database instances like other DB2‘s

Two Interfaces

Power Systems

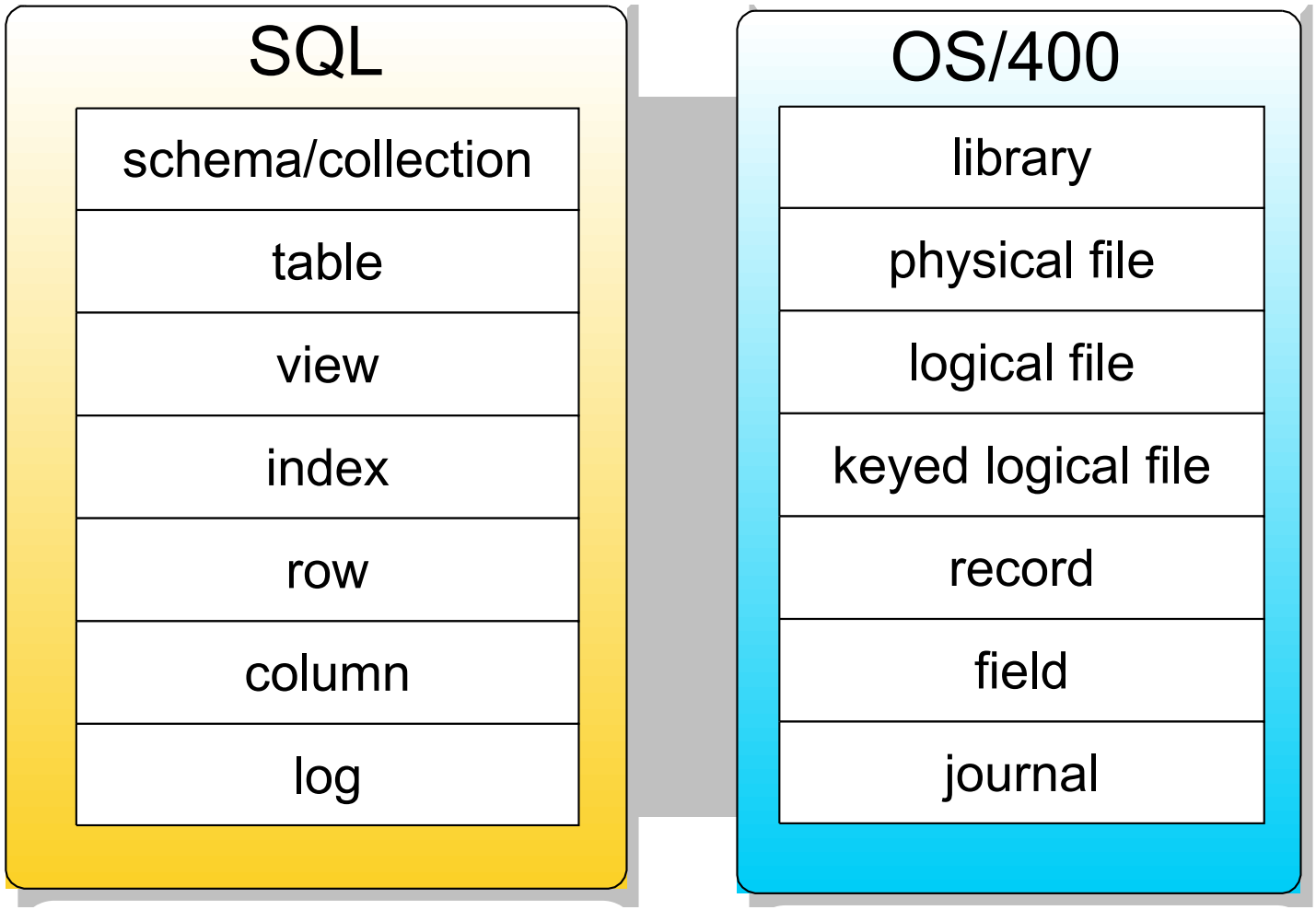






8.2.1 Native Database Implementation

Database Objects - Terminology



Terminology

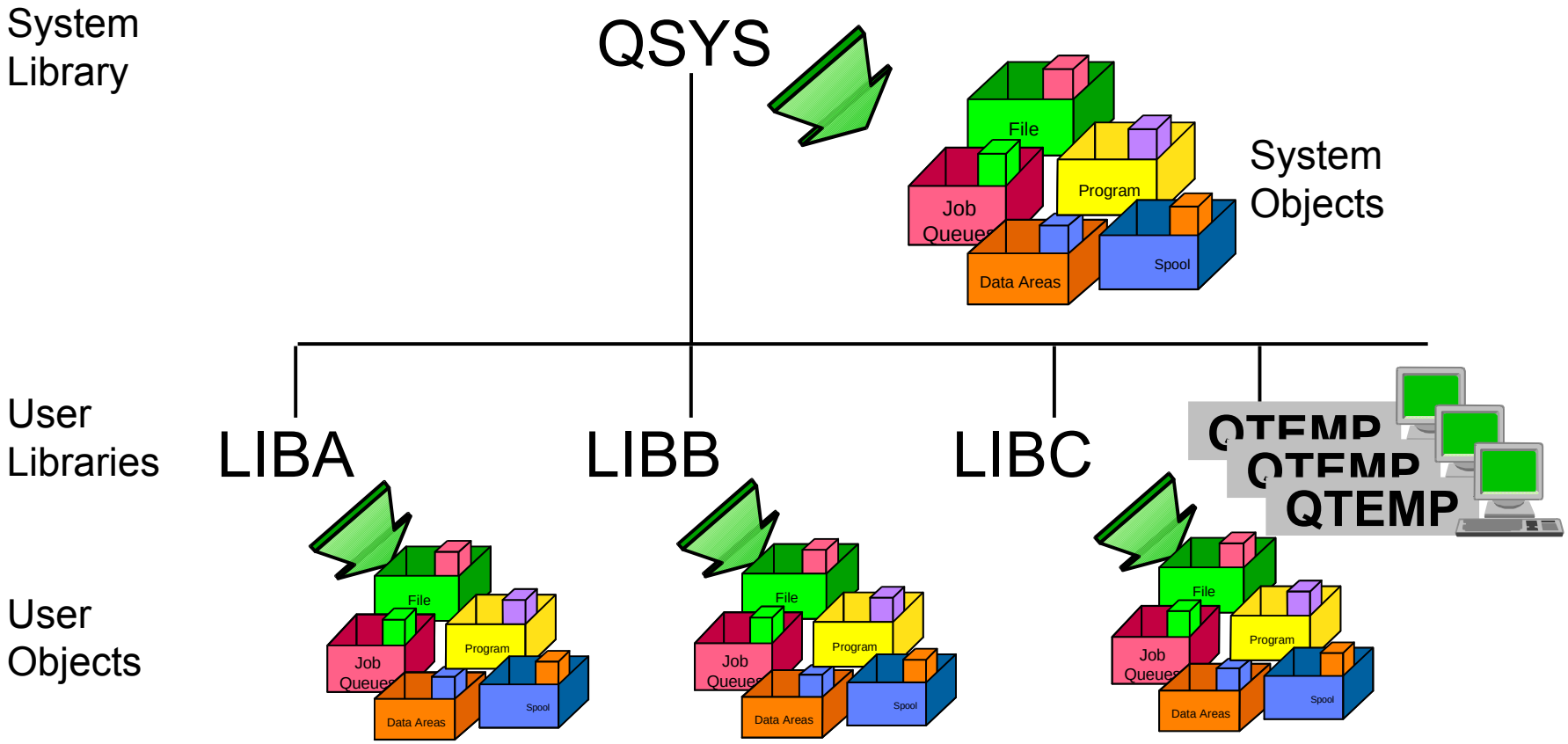
Power Systems

SQL versus traditional file access terminology

SQL Term	Traditional File Access Term
Collection. Consists of a library, a journal, a journal receiver, an SQL catalog, and an optional data dictionary. A collection groups related objects and allows you to find the objects by name.	**Library.** Groups related objects and allows you to find the objects by name.
Table. A set of columns and rows.	**Physical file.** A set of records.
Row. The horizontal part of a table containing a serial set of columns.	**Record.** A set of fields.
Column. The vertical part of a table of on data type.	**Field.** One of more bytes of related information of one data type.
View. A subset of columns and rows of one or more tables.	**Logical file.** A subset of fields and/or records of up to 32 physical files.
Index. A collection of data in the columns of a table, logically arranged in ascending or descending order.	A type of logical file.
Package. An object that contains control structures for SQL statements to be used by an application server.	**SQL Package.** Has the same meaning as the SQL term.
Catalog. A set of tables and views that contain information about tables, packages, views, indexes, and constraints.	No similar object. However, the Display File Description (DSPFD) and Display File Field Description (DSPFFD) commands provide some of the same information that querying an SQL catalog provides.

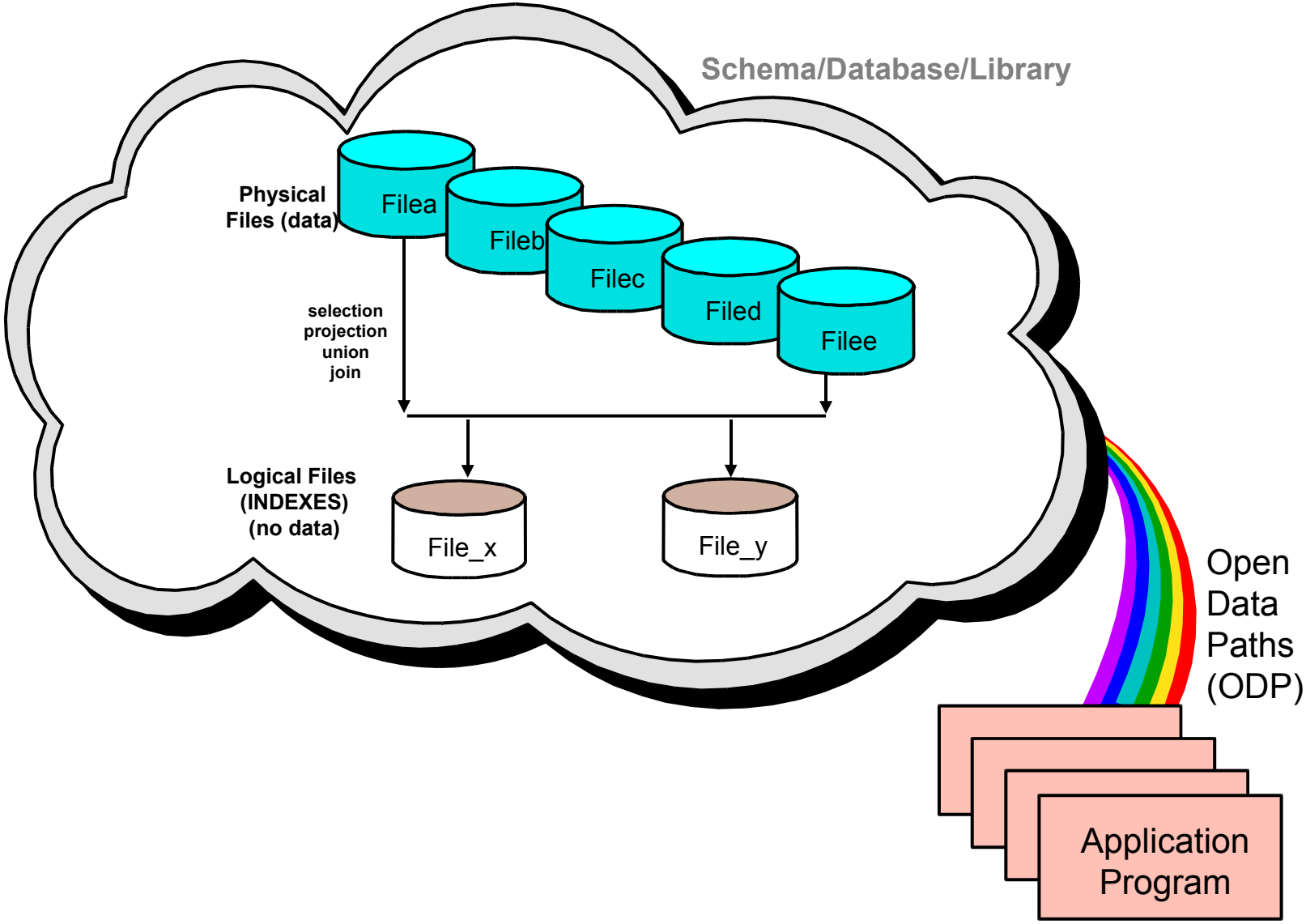
Native Library Structure

Power Systems



Physical/Logical Files

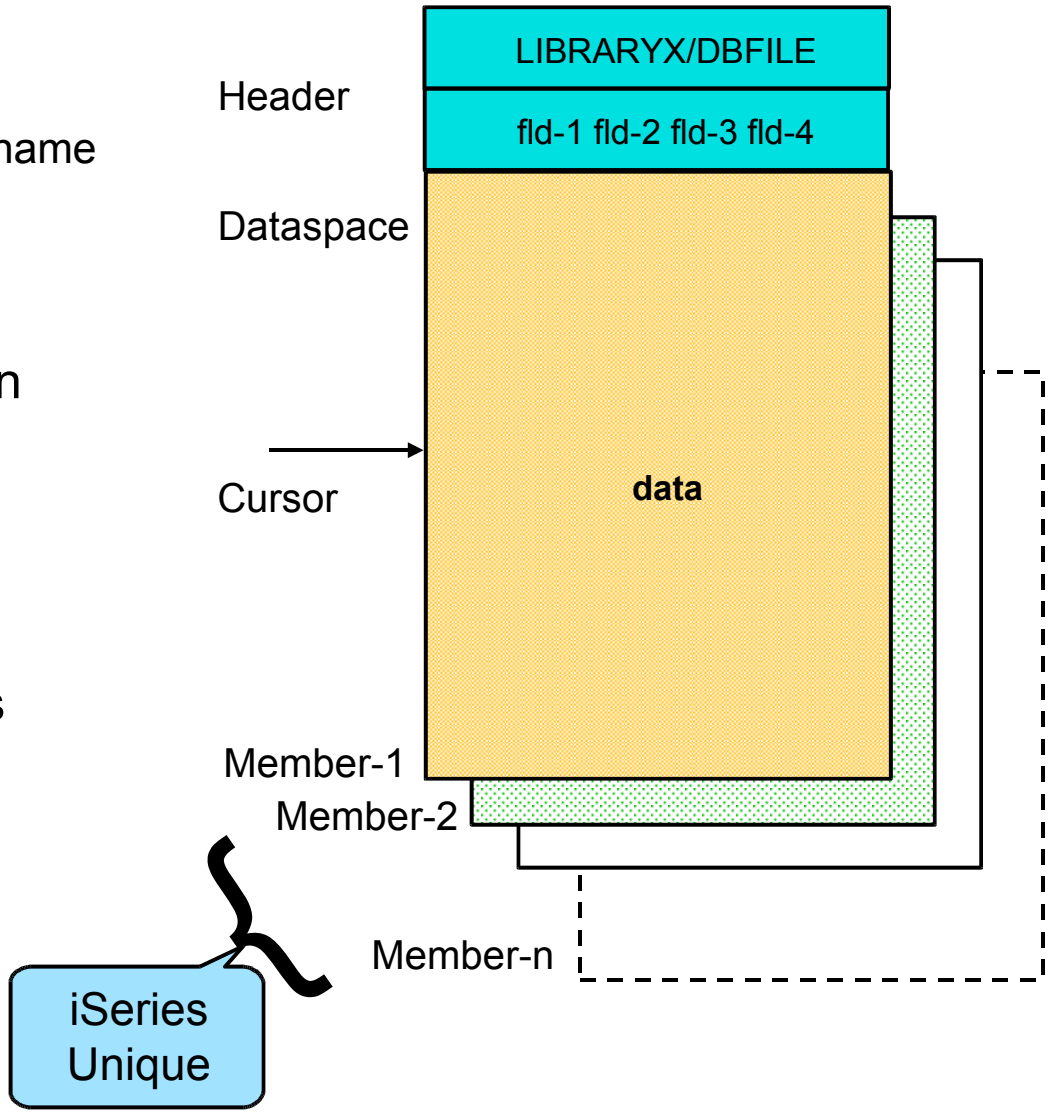
Power Systems



Physical File Structure

Power Systems

- File Header
  - File Description
    - Qualified File library/name
    - Owner
    - Object Authority
    - Number of records
  - Field Level Description
    - Field names
    - Attributes
- Dataspace
  - Actual Data
  - Multiple File Members
    - Not used by SQL
- Index Space
  - Index
- Cursor



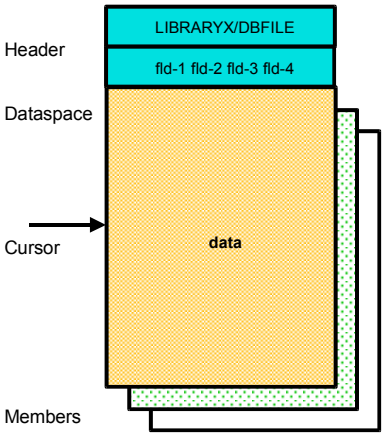
File Members

Power Systems

```
                              Display Physica
File . . . . . :   CUSTOMERS
Member . . . . . :   SECOND
Control  . . . . .
Find . . . . . .
*...+....1....+....2....+....3....+....4
918374Davon    A C249 North St LondonTX75
833245Lee      D S21B NW 135 StClay  NY13
123859Jenner   T PPO Box 79    BrotonVT05
345485James    S A3 Alpine Way Helen GA30
393457Peters   D A13 Myrtle Dr HectorNY14
323472Wang     C M208 Snow PassDenverCO80
889900Hoover   E J787 Lake Dr  SydneyMN56
423438HumphreyH W59 Archer Rd SutterCA95
693829Thomas   A N3 Dove CircleCasperWY82
593029WilliamsE D485 SE 2 Ave BangorTX75
192837Lee      F L5963 Oak St  ForestNY14
```

```
                        Display Physical File Member
File . . . . . :   CUSTOMERS          Library  . . . . :   ERPDATA
Member . . . . . :   CUSTCDT           Record . . . . . :   1
Control  . . . . .                    Column . . . . . :   1
Find . . . . . .
*...+....1....+....2....+....3....+....4....+....5....+....6
938472Henning G K4859 Elm Ave DallasTX752175000300370000000
839283Jones    B D21B NW 135 StClay  NY13041040010100000000
392859Vine     S SPO Box 79    BrotonVT05046070010439000000
938485Johnson J A3 Alpine Way Helen GA30545999923987500033
397267Tyron    W E13 Myrtle Dr HectorNY14841100010000000000
389572Stevens K L208 Snow PassDenverCO80226040010058750001
846283Alison   J S787 Lake Dr  Isle  MN56342500030010000000
475938Doe      J W59 Archer Rd SutterCA95685070020250000100
693829Thomas   A N3 Dove CircleCasperWY82609999920000000000
593029WilliamsE D485 SE 2 Ave DallasTX75218020010025000000
192837Lee      F L5963 Oak St  HectorNY14841070020489500000
583990Abraham M T392 Mill St  Isle  MN56342999930500000000
                                    ND OF DATA ******
```

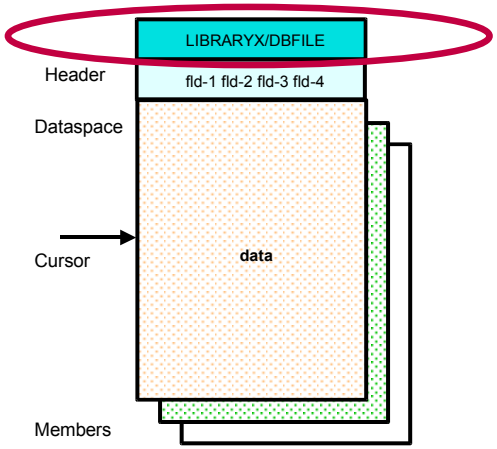
```
                        Display Physical File Member
File . . . . . :   CUSTOMERS          Library  . . . . :   ERPDATA
Member . . . . . :   THIRD             Record . . . . . :   1
Control  . . . . .                    Column . . . . . :   1
Find . . . . . .
*...+....1....+....2....+....3....+....4....+....5....+....6
323472Wang     C M208 Snow PassDenverCO80226040010058750001
889900Hoover   E J787 Lake Dr  SydneyMN56342500030010000000
423438HumphreyH W59 Archer Rd SutterCA95685070020250000100
693829Thomas   A N3 Dove CircleCasperWY82609999920000000000
593029WilliamsE D485 SE 2 Ave BangorTX75218020010025000000
192837Lee      F L5963 Oak St  ForestNY14841070020489500000
583990Abraham M T392 Mill St  IslandMN56342999930500000000
                        ****** END OF DATA ******
```



File Information (1)

Power Systems

- File Description Header
  - Explicit File Name
    - Library Name
    - File Name (10 characters)
      - Alternative File Name (20 characters)
  - File Type
    - Data -
    - Source - program code
- Database Attributes
  - File Definitions
    - Externally Described
      - Column (or Field) Level Specifications
      - Field information stored with file
    - Program Described
      - Row (or Record) level Description
      - Field information within program(s)
  - File Creation Date
  - File Size Limits
    - Record Length
    - Initial records
    - Size of increment
    - Number of increments

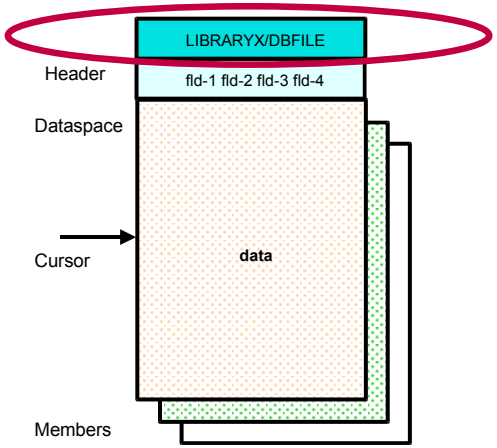


DSPFD Command

File Information (2)

Power Systems

- Database Attributes
  - Number of members
    - Maximum
    - Current
- Reuse Deleted Records
- Force Write to Disk
- File Activity
  - Number
    - Opens/Closes
    - Update/Delete Operations
  - Date of Last
    - Change
    - Save
    - Restore
- File Member Information
  - Name/Creation Date
  - Last Update (date/time)
  - Number or Records

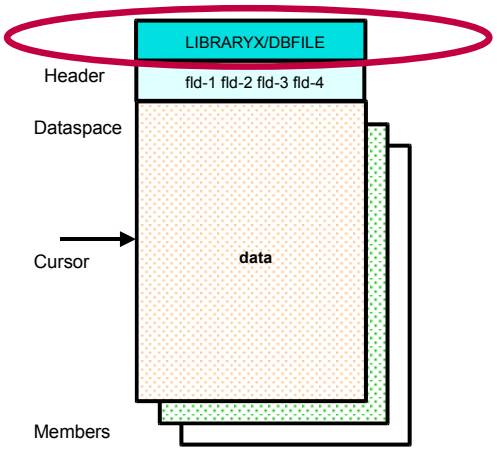


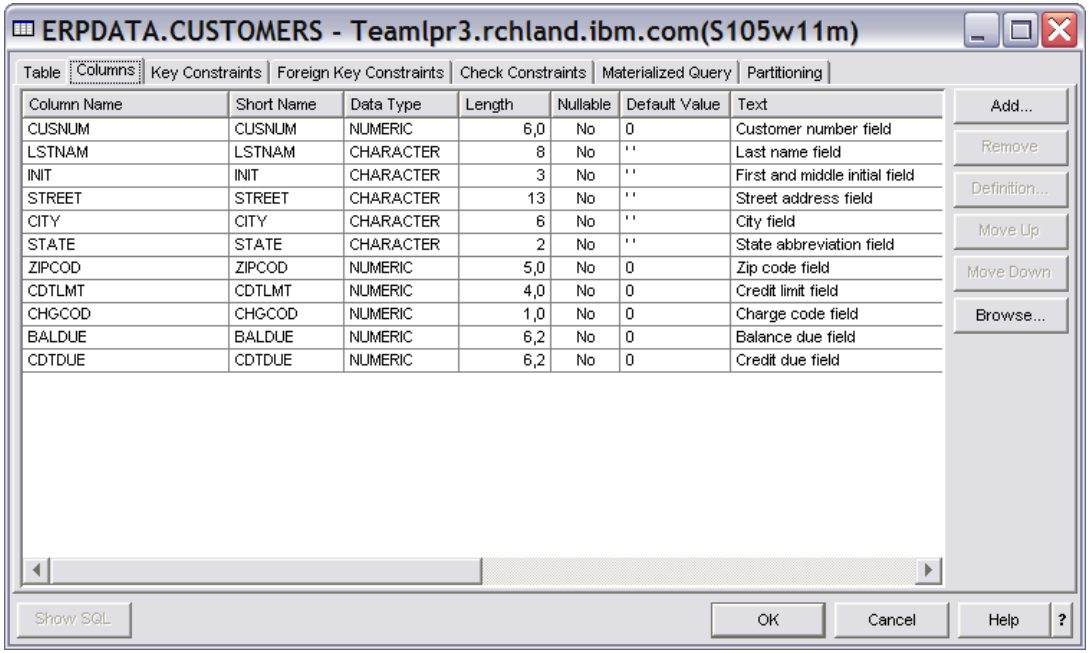
DSPFD Command

Feld Field Description - Record Information

Power Systems

- Columns (Field) Information
  - –Field Name
  - –Data Type
    - •Character
    - •Numeric
    - •Binary
  - –Length
  - –Buffer Positions
  - –Usage
    - •Input/Output
  - –Column Heading





DSPFFD Command (Green Screen)

Create a Physical File with DDS

Power Systems

Field Reference File :

```
          R FLDREFP
            PERSNR            5S    0      COLHDG('Personal-' 'nummer')
            NAME            1000           VARLEN(36)
                                           COLHDG('Name')
            GEBDAT             L           COLHDG('Geburts-' 'datum')
            ABTLG             4S   0       COLHDG('Abteilung')
            GEHALT            7   2        COLHDG('Gehalt')  ALWNULL
            ...
```

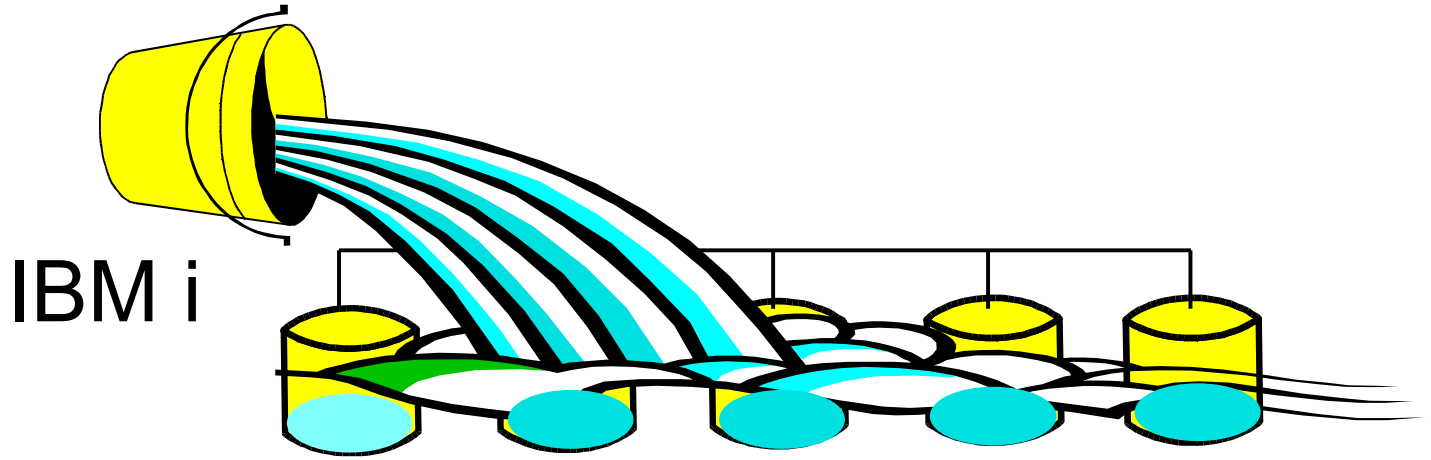
Physical File Description with DDS :

```
                                        REF(FLDREFP)
        R  PERSTPF1
            PERSNR          R
            NAME            R
            GEBDAT          R
            ABTLG           R           ALWNULL
            GEHALT          R
```

8.2.2 IBM i Data Retrieval

Disk Storage Management

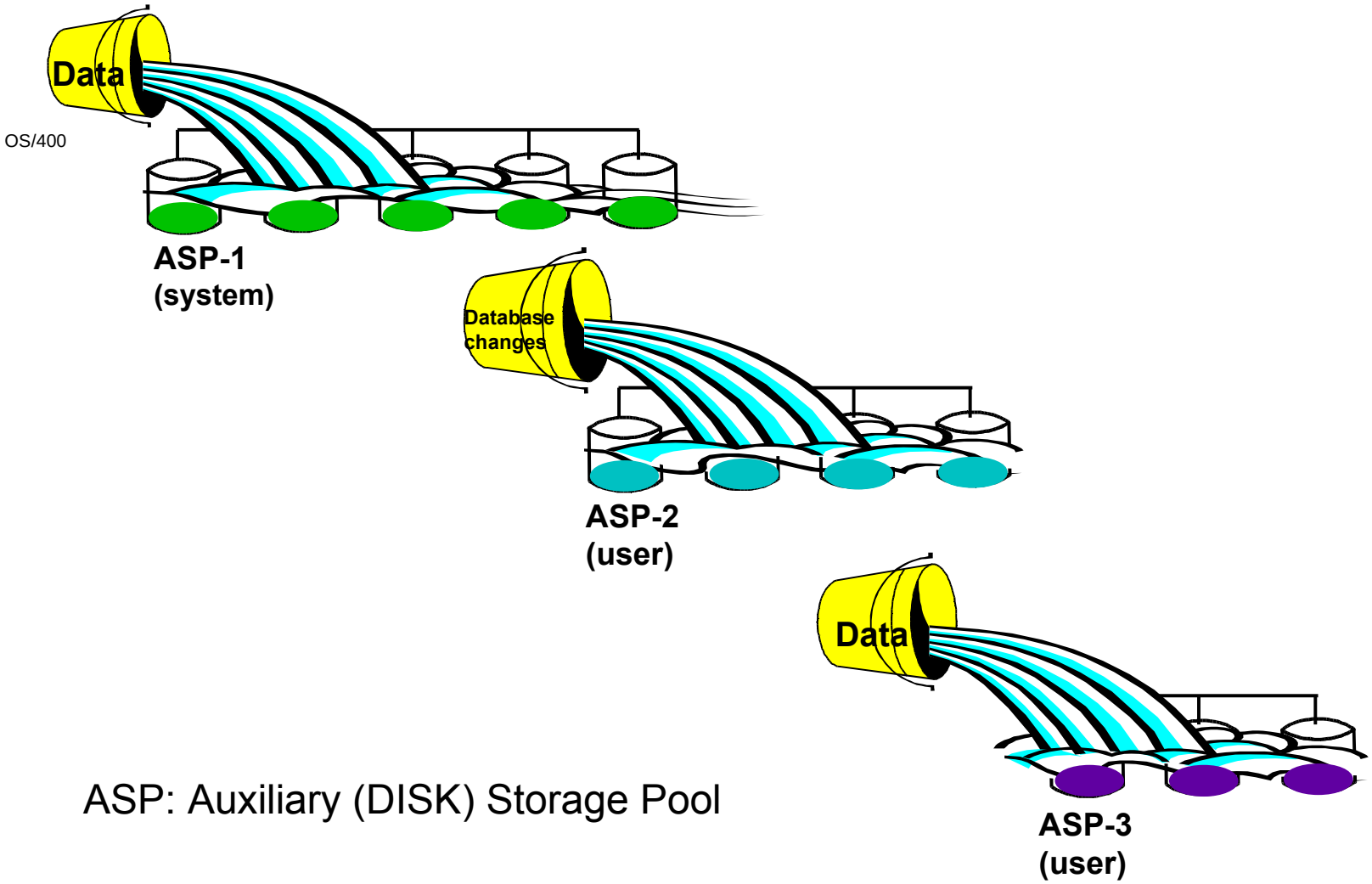
Power Systems



- All data is spread across available disk arms
  - Optimum performance - automatically
- Not all information is necessarily contiguous
  - Improved performance
    - Balanced disk arm utilization
- Optional rebalancing
  - space/arm utilization
- Information accessed by name not hardware address
- Minimal Database Administration

Multiple ASPs

Power Systems

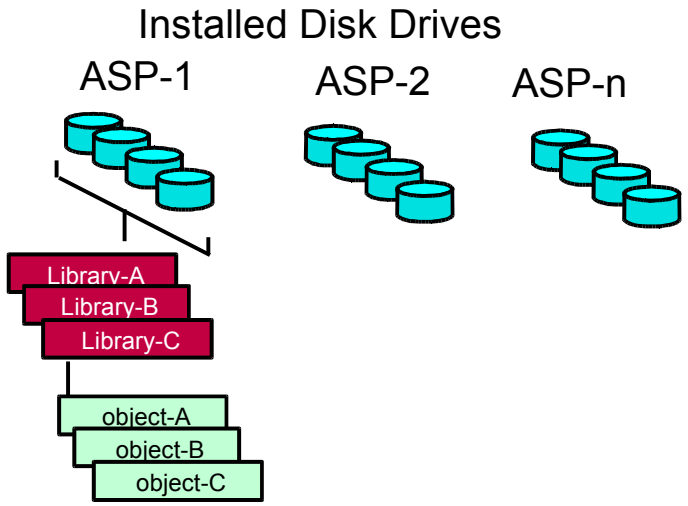


ASP: Auxiliary (DISK) Storage Pool

Impact on Data Retrieval

Power Systems

- An ASP can (does) span multiple disks
  - Contains Multiple Libraries
- Data Objects not bounded by Disk Volumes
  - Data may span multiple disk drives
- Application Database
  - Multiple Libraries
    - Multiple Objects
      - Physical Files
      - Logical Files
      - Data Areas
      - Data Queues
      - And so forth
  - IFS Objects
    - Multiple directories
      - Sub-directories



Physical Files (Data)

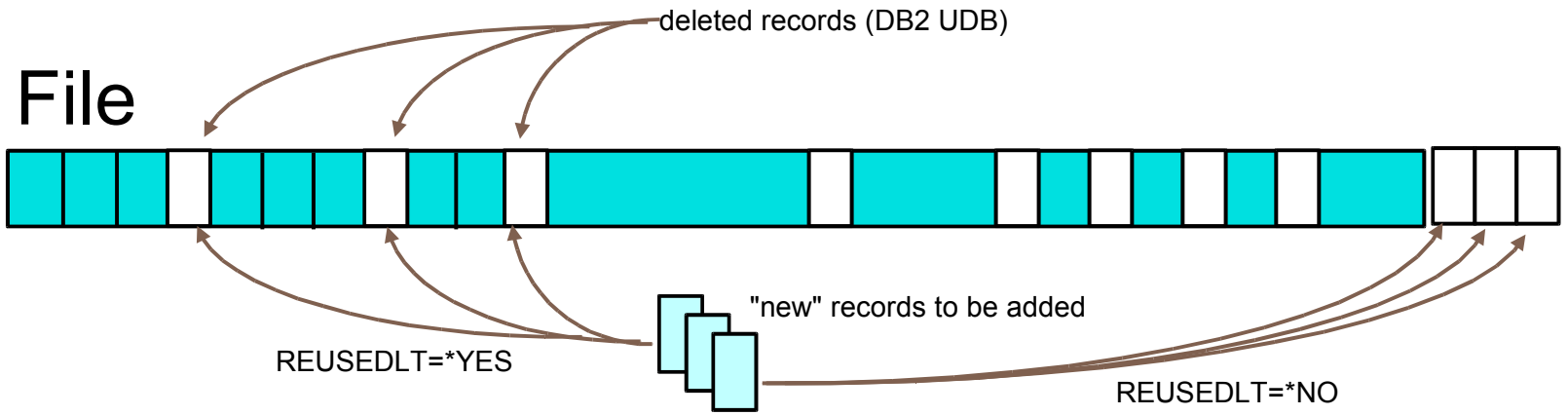
Power Systems

- Actual Information Stored in DB2 UDB Physical Files
  - Type *DATA (versus *SOURCE)
- Specify
  - File Name
    - <library-name>/<file-name>
      - Data held in <member-name>
- Number of Members in file
  - Data Format
    - Record (row) Length
    - Field (column) Organization
      - Type
        Character
        Decimal
        Binary and so forth
      - Length
  - Reuse Deleted Records
    - REUSEDLT = *YES/*NO
  - Records to force a write
    - FRCRATIO=*NONE/<value>

Deleted Records

Power Systems

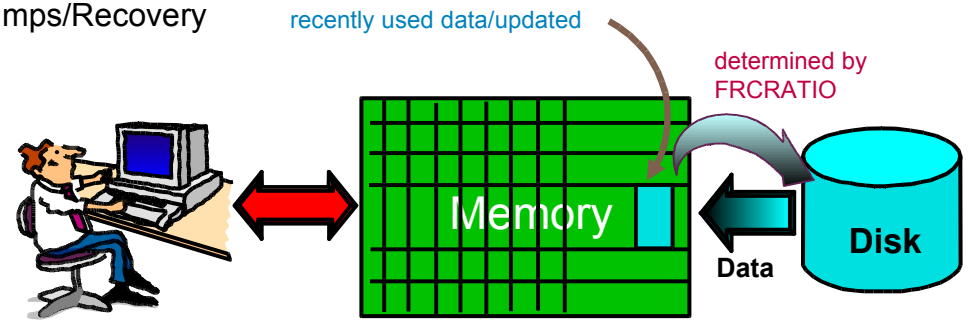
- Through DBMS Support (DB2 UDB)
  - REUSEDLT = *YES
    - Flagged by DB2 UDB
      - Records not made available to applications on *READ
    - Parameter REUSEDLT in CRTPF/CHGPF
      - *YES
        Space freed by deletion of a record is available for additions (new records)
      - *NO
        Space is not freed when a record is flagged for deletion
  - Space "freed" by record deletion
    - File "compression" possible
      - Reorganize Physical File Mbr (RGZPFM command)
- Reduces File size (dependent on number of deletions)



Force Records to Disk

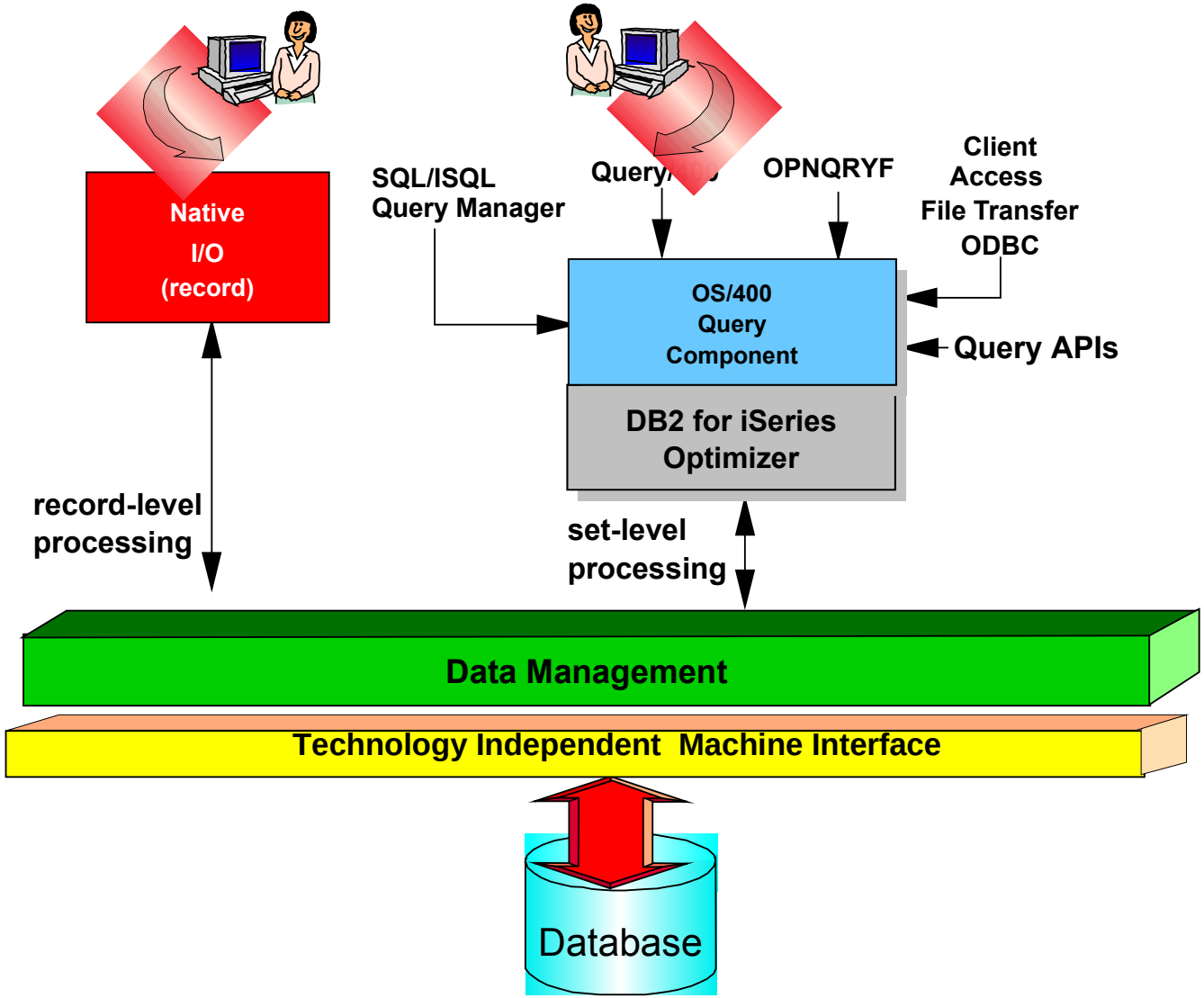
Power Systems

- Arises from Single Level Storage Implementation
  - Frequently access pages tend to remain in memory
    - Applies to ALL OBJECTS
- Override for Physical Files
  - Data written to Disk determined by parameter FRCRATIO
    - *NONE ..... System Managed
    - <value>..... Number of Records
      - Added/Updated
- Consider Impact of Sudden Power Outage!
  - Memory is Volatile
    - Data loss?
      - Battery Failure
      - Memory Dumps/Recovery
- At next IPL



Accessing the Database

Power Systems

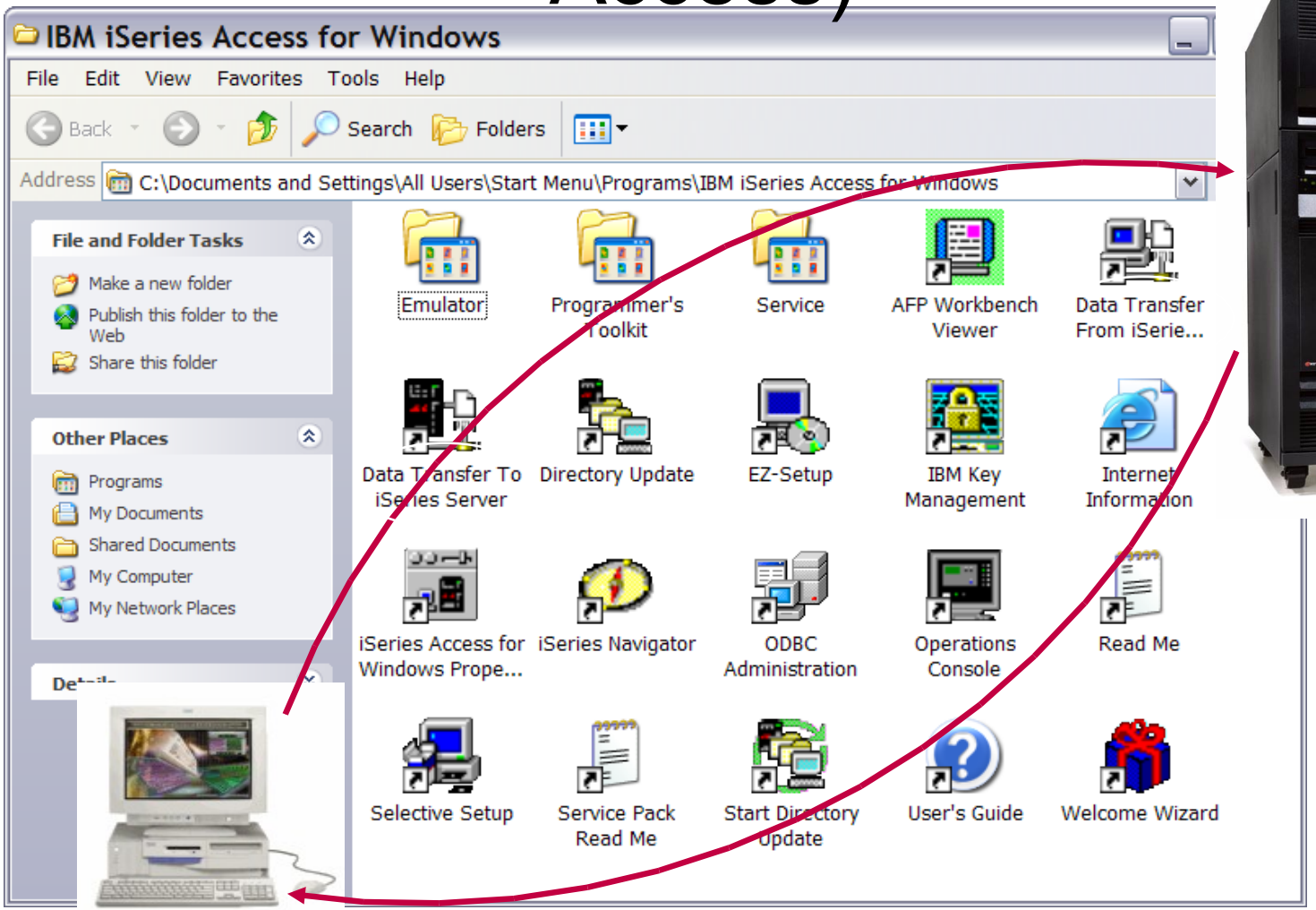


Data Access

- Software interacting with DB2 UDB for i5/OS
  - Applications
    - High Level Languages
    - Application Packages
  - iSeries Query Tools
    - GUI
      - iSeries Access for Windows (5722-XE1)
      - iSeries Navigator
    - Character-based interface
      - Query Manager (5722-QU1)
      - DB2 Query Manager (5722-ST1)
      - OS/400 Commands
        Copy File (CPYF)
        Create Duplicate Object (CRTDUPOBJ)
        Display File (DSPPFM)
    - Non-IBM Tools

File Transfer (iSeries Access)

Power Systems



More information through Hands-on Lab Exercises!

CPYF

Power Systems

```
                          Copy File (CPYF)

 Type choices, press Enter.


 From file  . . . . . . . . . . . .  <input-file>   Name
   Library  . . . . . . . . . . . .    <library>    Name, *LIBL, *CURLIB
 To file  . . . . . . . . . . . . .  <output-file>  Name, *PRINT
   Library  . . . . . . . . . . . .    <library>    Name, *LIBL, *CURLIB
 From member  . . . . . . . . . . .   *FIRST        Name, generic*, *FIRST,
*ALL
 To member or label . . . . . . . .   *FIRST        Name, *FIRST, *FROMMBR
 Replace or add records . . . . . .   *NONE         *NONE, *ADD, *REPLACE...
 Create file  . . . . . . . . . . .   *NO           *NO, *YES
 Print format . . . . . . . . . . .   *CHAR         *CHAR, *HEX
```

8.2.3 To DBA or not to DBA?

Follow-On Advantages of i5/OS Architecture

Database Administrator Tasks

Power Systems

- Traditionally, a DBA is a database administrator performing tasks such as...
  - Installing the RDBMS
  - Configuring the RDBMS
  - Starting and Stopping the RDBMS
  - Allocation of disk space for database objects
  - Partitioning and balance of disk drives
  - Reorganizing / rebalancing of system index structures
  - Maintaining statistics used in query optimization
  - Logical and physical data modeling
  - Determining and implementing indexing strategies
  - User administration and security
  - Monitoring and tuning database access and processes

The DBA

- Traditionally, DBAs are separated from the application development community
  - Technically different roles and responsibilities
  - Organizationally different roles and responsibilities

What is a DBA …?

Power Systems

What's the difference between...

- Database Administrator
- Database Architect
- Database Analyst
- Data Steward
- Data Owner

Why are they needed...

- Technical reasons
- Organization reasons
- Philosophical reasons
- Historical reasons

Database Skill/Tasks

Power Systems

UNIX/Wintel DBA Tasks	DB2 UDB for i5/OS
Manage DASD Space Allocation	Completely Automated
Review Table Space Allocations and Extents	Completely Automated
Review and Balance Indexes	Completely Automated
Application Rebinding	Completely Automated
Maintain Database Integrity	Completely Automated
Update Database Statistics	Completely Automated
Synchronized OS and DB User Security	Completely Automated
Reload Data for Hardware and Software Upgrades	Completely Automated
Load Data into Data Base	Integrated Utility (Parallel)
Build and Manage DB Backup and Recovery	Integrated GUI
Create and Review Indexes for Tables	Integrated GUI
Performance Analysis and Tuning (DB and System)	Integrated GUI
Create and Maintain DB Schema	Integrated GUI, DB2 OLAP, and 3rd Party Tools (ERwin,S-Designer)
Automated DB Performance Profiling	insure/SQL
Advanced DB Performance Analysis and Tuning	insure/SQL
Data Replication and Consolidation	Multiple IBM and 3rd Party Products

Integrated GUI - iSeries Navigator
insure/SQL by Centerfield Technologies

Do you need a DBA on IBM i? Maybe...

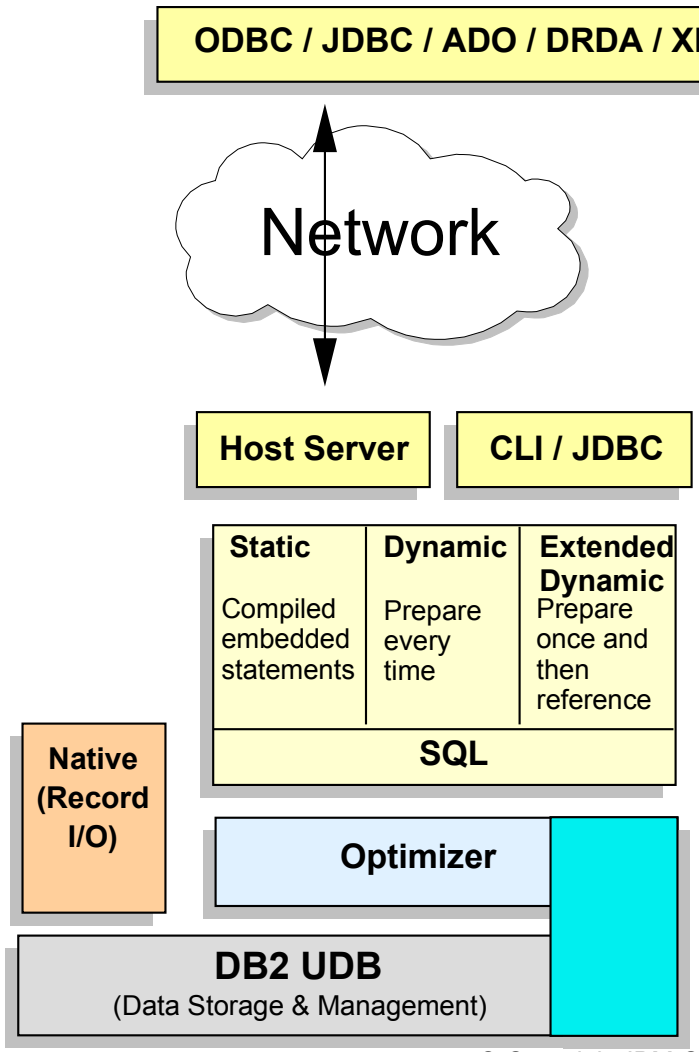
Consider...

- Database *Analyst* or *Architect* (not administrator)
- Technical issues of moving to a data-centric world
  - Business logic moving into the database
  - More data, used in more places, by more clients
- Organizational issues
  - Division of work
  - Level and type of knowledge required is specialized

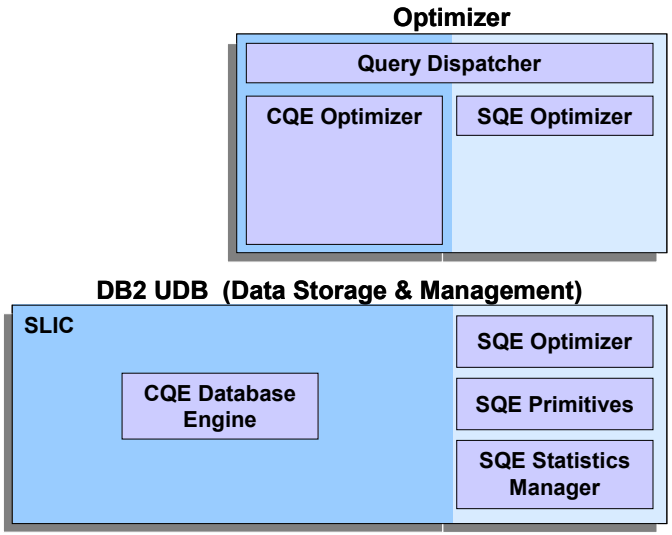
8.2.4 Performance

System i Database Architecture

Power Systems



The "classic" optimizer and database engine merged to form the SQL Query Engine and much of the work was moved to SLIC.



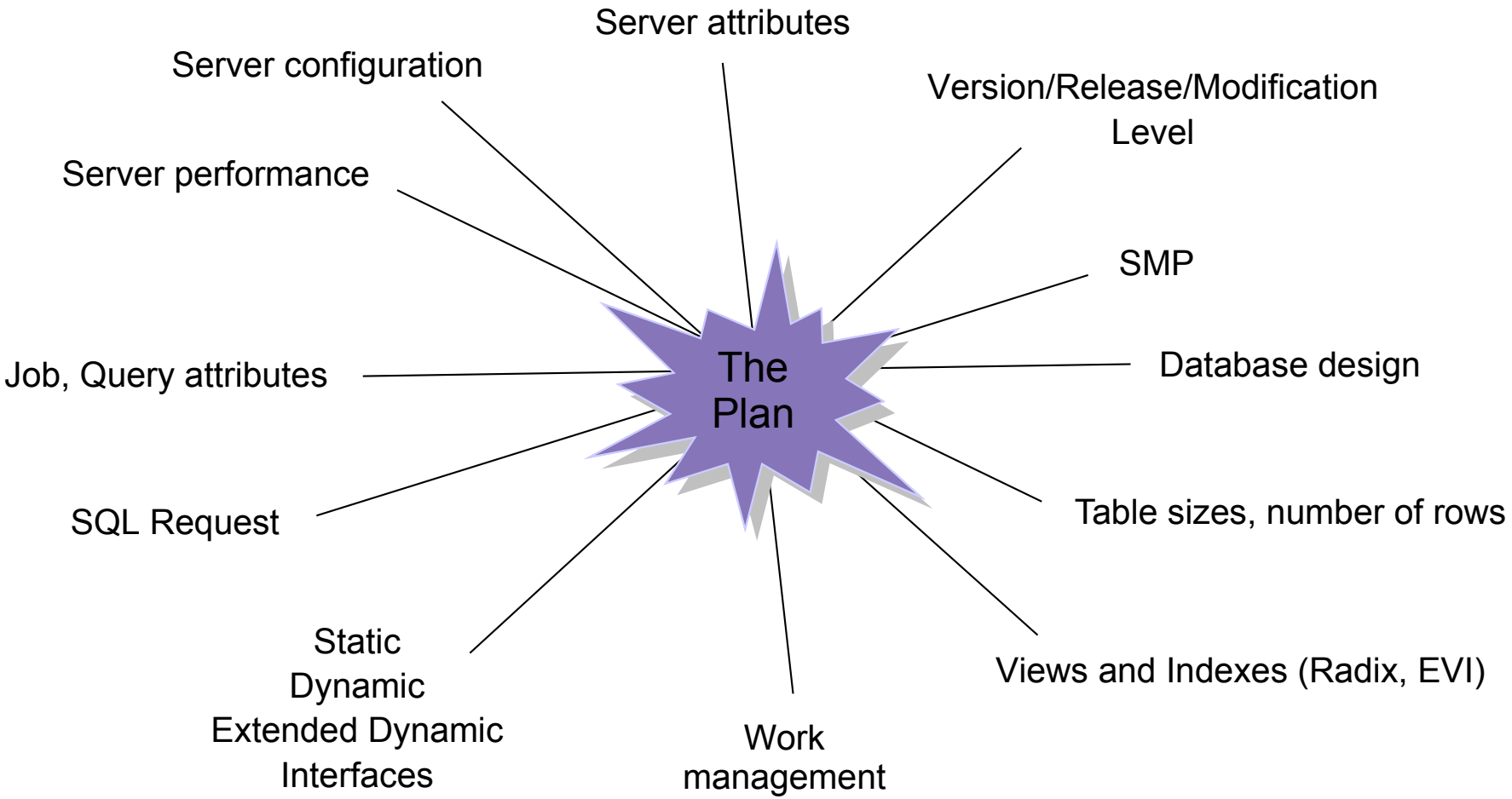
Five Factors of DB Performance

- Workload
  - OLTP, OLAP, ad-hoc queries, system commands
    - Can fluctuate drastically
- Throughput
  - Overall capability to process data
    - CPU speed, I/O speed, parallel capabilities
- Resources
  - Hardware- and software tools
    - Database kernel, disk storage, RAM, cache, microcode
- Optimization
  - Query optimization is primary accomplished internal
  - SQL formulations, index strategy and database parameter
- Contention
  - Condition, where two or more components are attempting to use a single resource in a conflict way
  - As contention increases, throughput decreases

Therefore, database performance can be defined as the optimization of resource use to increase throughput and minimize contention, enabling the largest possible workload to be proceed.

* Source: Craig S. Mullins, Database Administration – The Complete Guide to Practices and Procedures

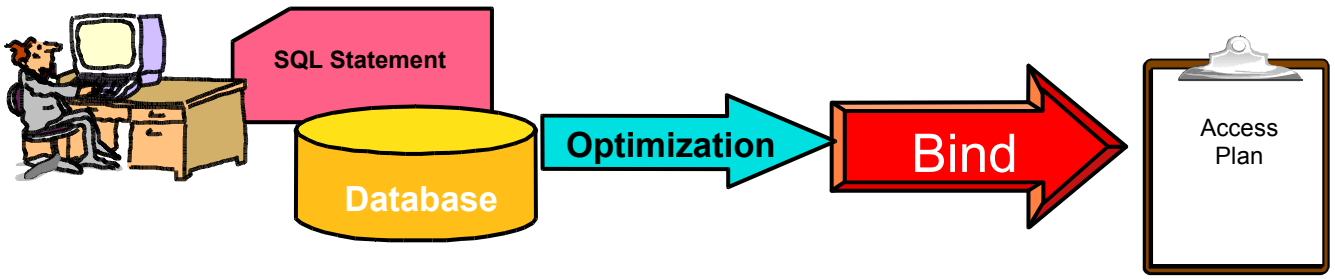
Various Factors of Optimization



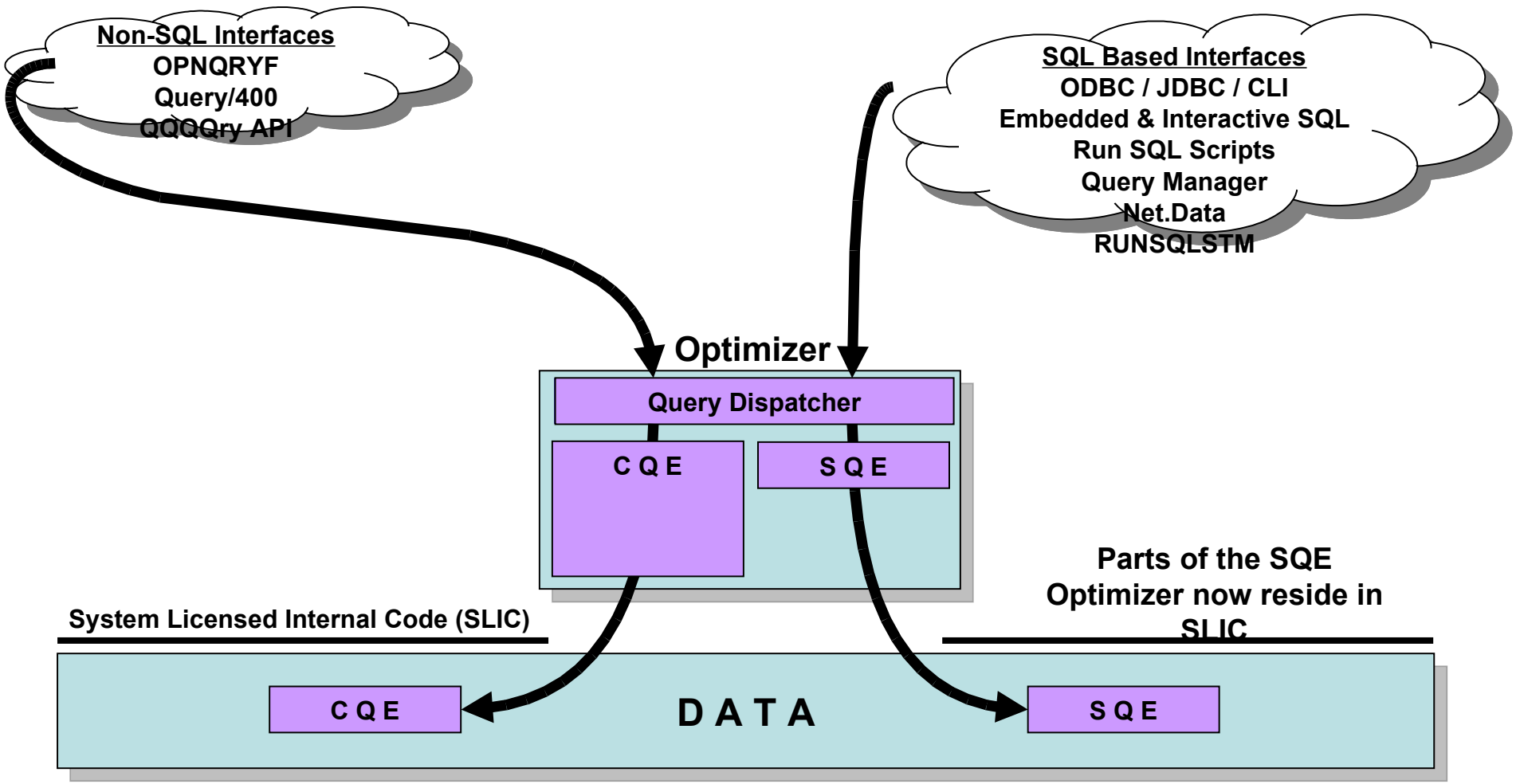
SQL Optimizer

Power Systems

- Why?
  - Improved Database performance
    - Identify technique used to implement query
    - Selects most efficient technique
- Access Plan
  - Internal structure defining method to process SQL statement
  - Created during SQL parse, syntax check
- Optimal access method selection based on
  - Implementation cost
    - SQL Statement
    - Current state of the database
      - Indexes/File Sizes



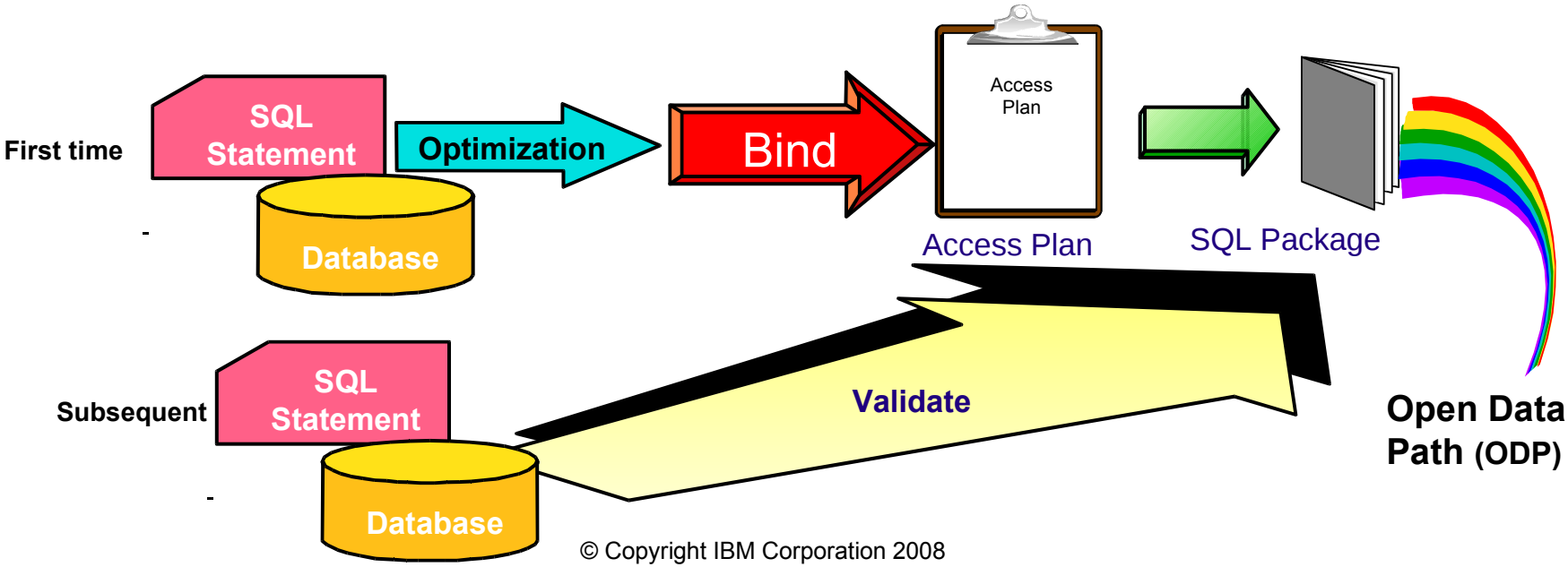
Query Dispatcher



SQL Packages

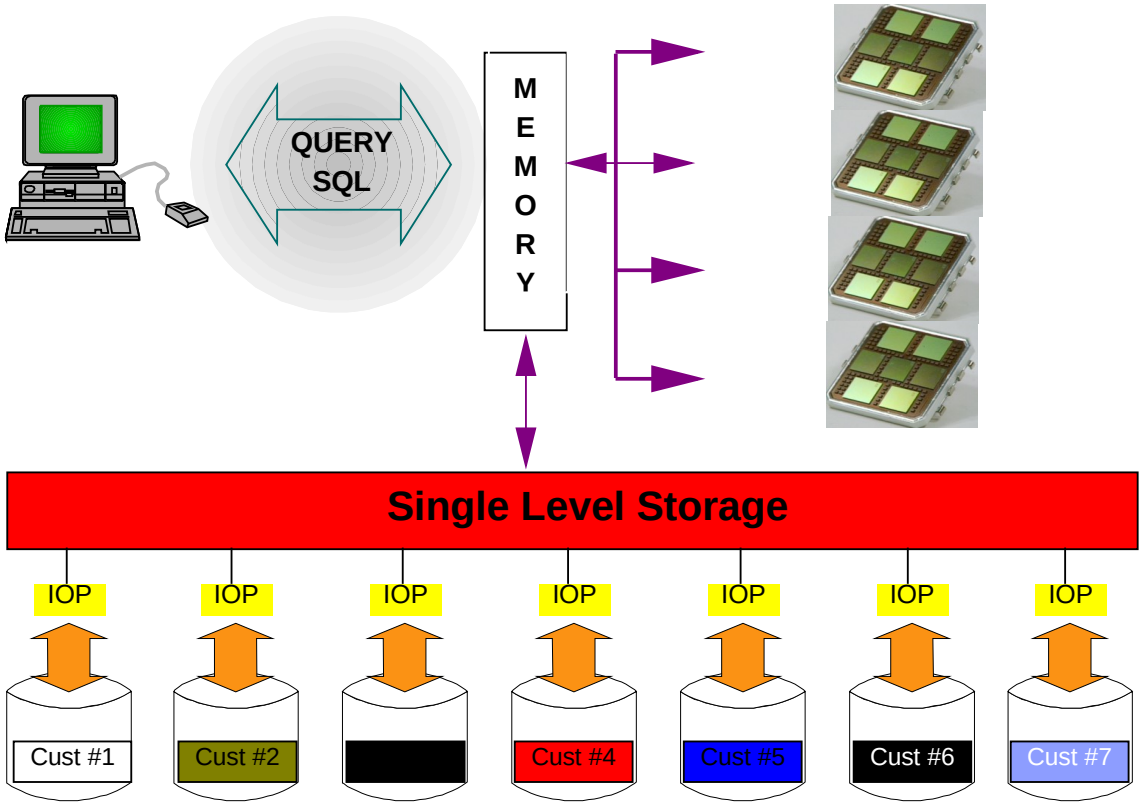
Power Systems

- IBM i Extended Dynamic SQL Support
  - Control structures used to execute SQL statements
  - Access Plans stored in *SQLPKG object
  - Improves Performance
  - IBM i commands : CRTSQLPKG, DLTSQLPKG
  - Dynamic re-optimization
    - File size
    - New/Deleted indexes



Parallelism

Power Systems



- Parallel table scan
- Parallel index scan
- Parallel hash join
- Parallel hash group by
- Parallel index build
- Parallel Data Load (Import PTF)
- Parallel index ANDing/ORing of dynamic bit maps
- Parallel Data Load (Export PTF)
- Parallel data load
- Parallel index maintenance
- Parallel Encoded Vector Index

Indexing

- Binary Radix Tree
- Bitmap Index
- Encoded Vector Index
- When will be indexes created?

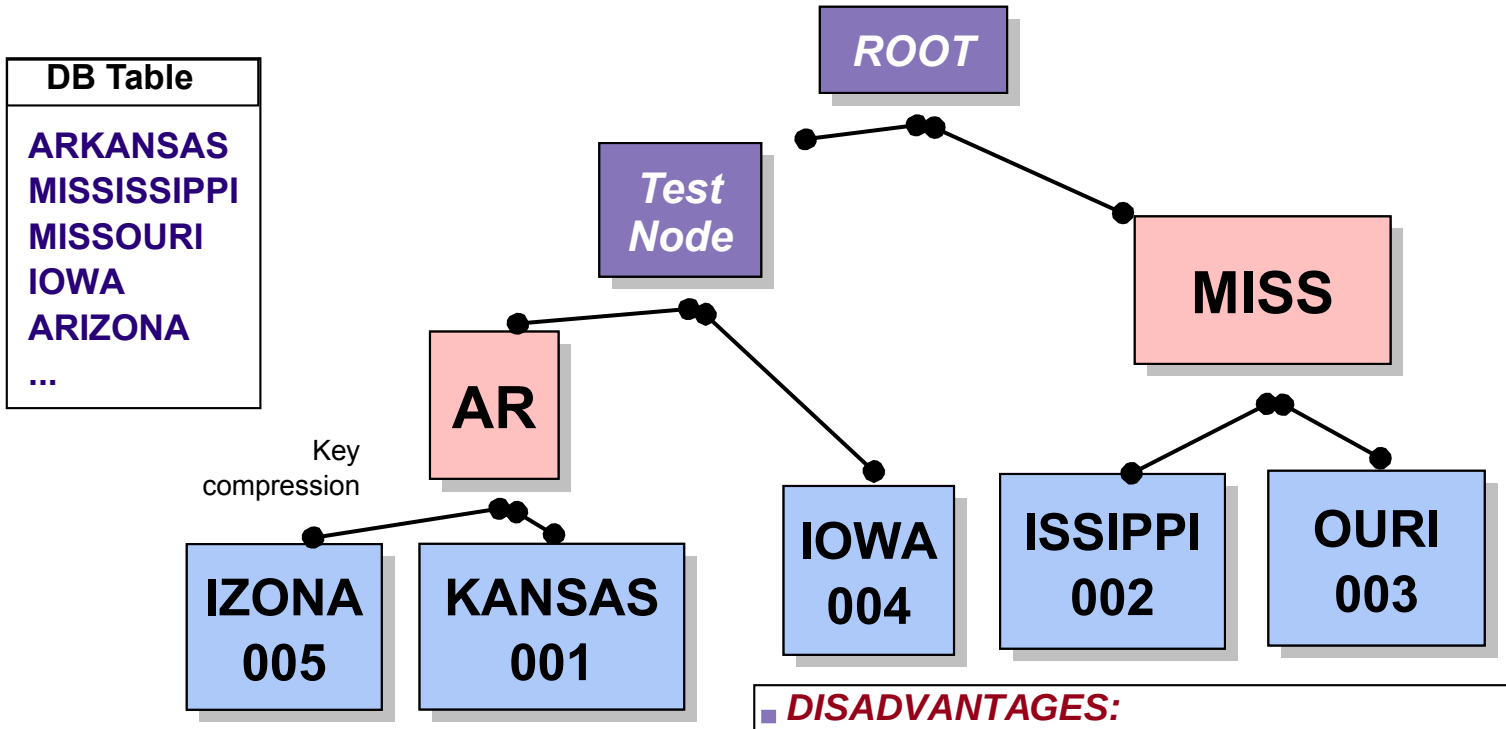
Binary Radix Index

Power Systems

- Key values are compressed
  - Common patterns stored once
  - Unique portion stored in "leaf" pages
  - Positive impact on size and depth of the "tree"
- Algorithm used to find values
  - Binary search
  - Very efficient process to find a unique value or small range of values
  - Modified to fit the data structure
  - Used to materialize a bitmap or relative record number (RRN)
- Maintenance
  - Index data is automatically spread across all available disk units
  - Tree is automatically "rebalanced" to maintain an efficient structure
  - No "index reorganization" required

Binary Radix Index

Power Systems

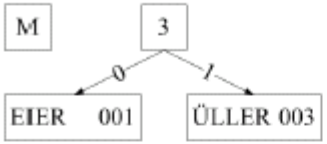


- ***ADVANTAGES:***
  - Quick access to a single key value (million-entry index, on average, only 20 tests)
  - Also efficient for small, selected range of key values (low cardinality)

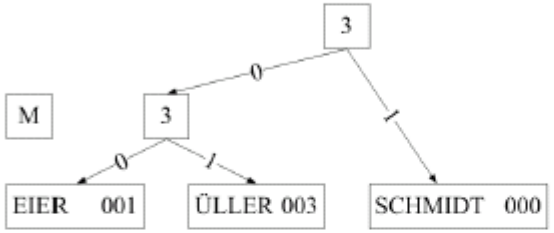
- ***DISADVANTAGES:***
  - Table rows retrieved in order of key values (not physical order) which equates to many ***RANDOM*** I/Os when selecting a large number of keys (high cardinality)
  - No way to predict which physical index pages are next when traversing the index for large number of key values

1. Add Meier 001:

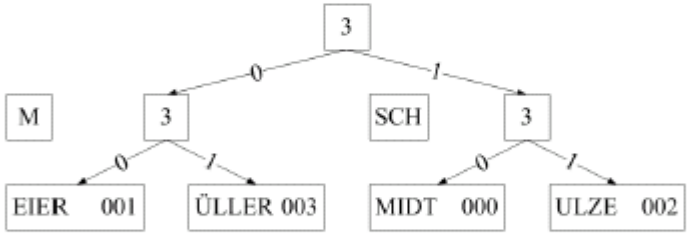
2. Add Müller 003:



3. Add Schmidt 000:



4. Add Schulze 002:



5. Add Zuse 004:

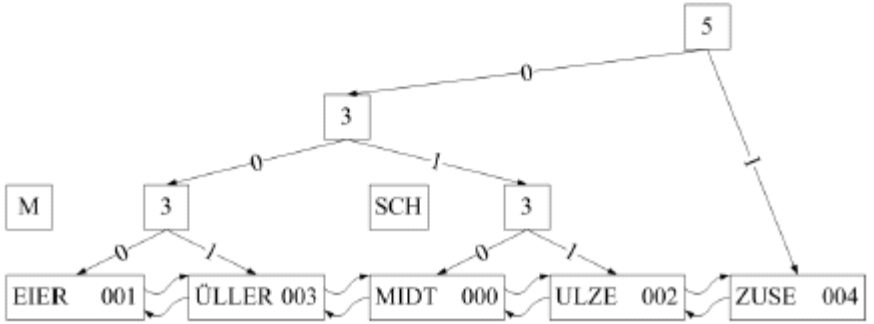
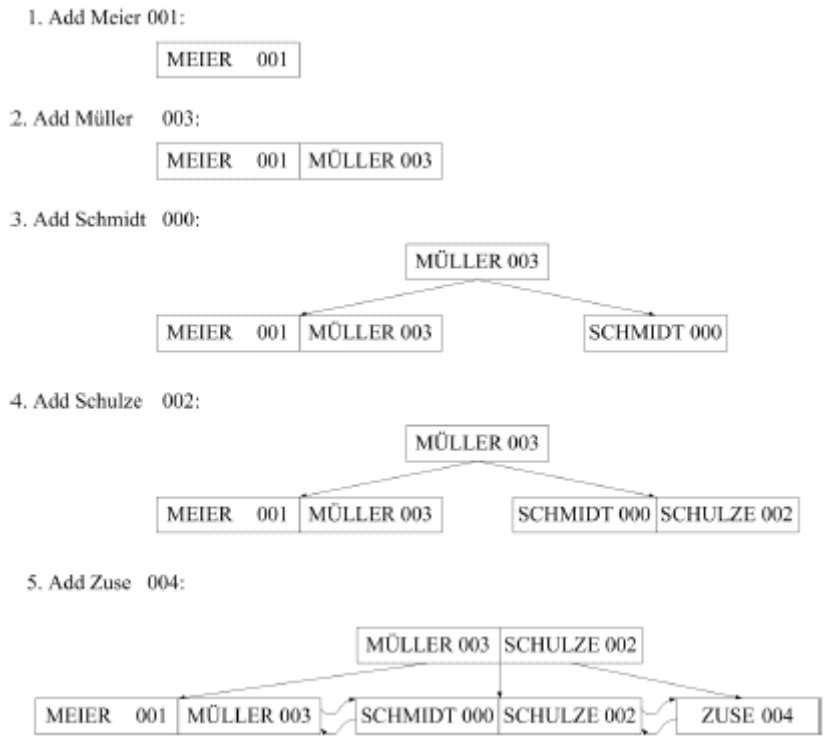


Abbildung 13: Aufbau binärer Radix-Baum

Vergleich zu B* Baum

Power Systems



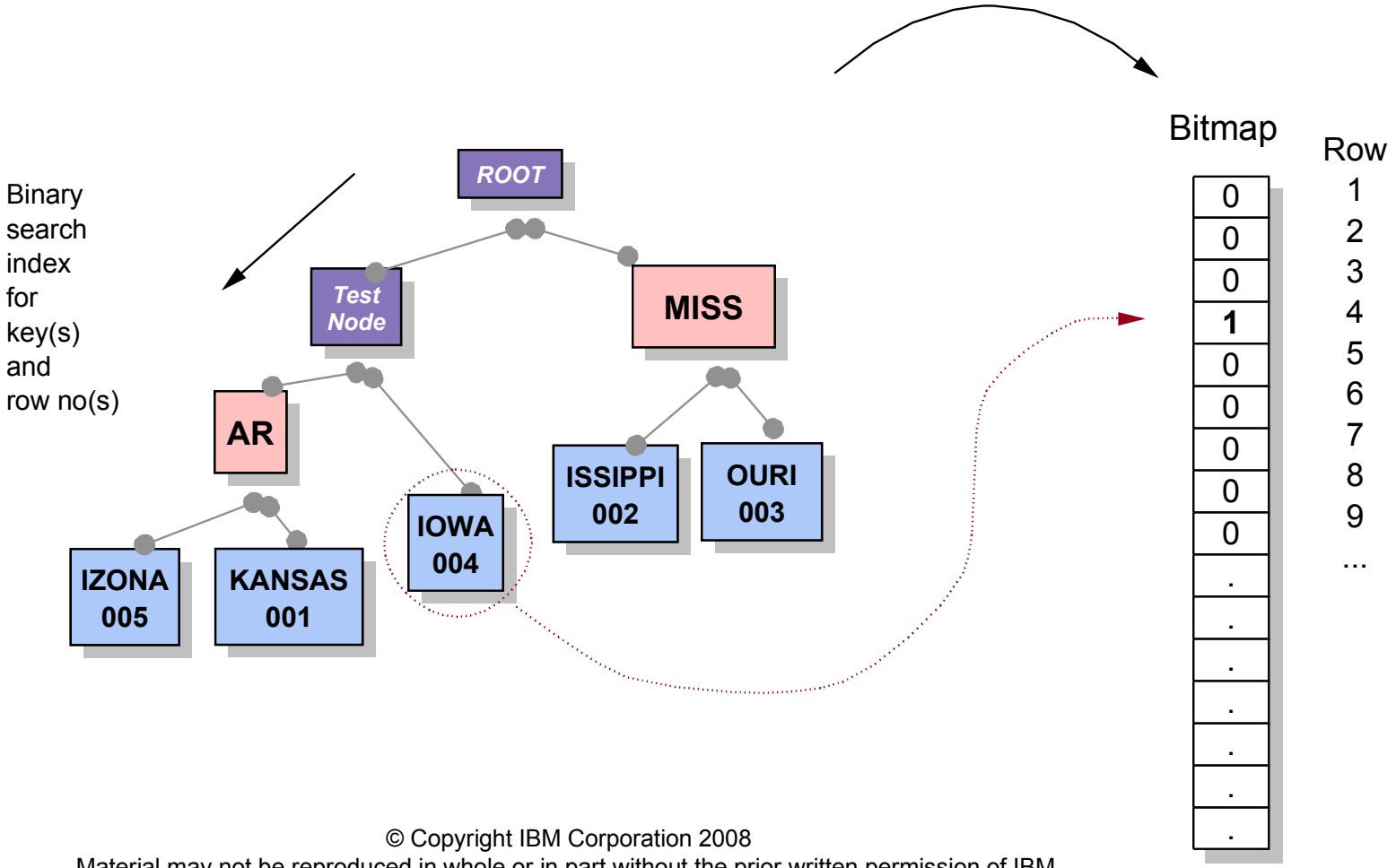
- Baumhöhe:
  - Der Radix-Baum ist von Natur aus nicht balanciert, wogegen beim B*-Baum alle Blätter den gleichen Abstand zur Wurzel haben
- Aufbau:
  - Während beim Radix-Baum lediglich ein Bit an einer vorbestimmten Stelle eines Bitstrings auf 0 oder 1 geprüft wird, muss beim B*-Baum jede Page, deren Größe von der Ordnung k des Baumes abh¨angt, von links nach rechts nach einer bestimmten Relation (z. B. <=) geprüft werden.
- Storage:
  - B* : jeder Knoten eine Page
  - Radix: Page enthält Teilbäume

Bitmap Index

Power Systems

Given an index on table EMPLOYEE keyed on STATE...

...WHERE STATE = 'Iowa'...



Encoded Vector Index (EVI)

- Index object for delivering fast data access in analytical query and reporting environments
  - Advanced technology from IBM Research, that is a variation on bitmap indexing
  - Complement to radix index (keyed logical file or SQL index)
  - Easy to access data statistics improve query optimizer decision making
  - Used to materialize a bitmap or relative record number (RRN) list
  - Easy to maintain

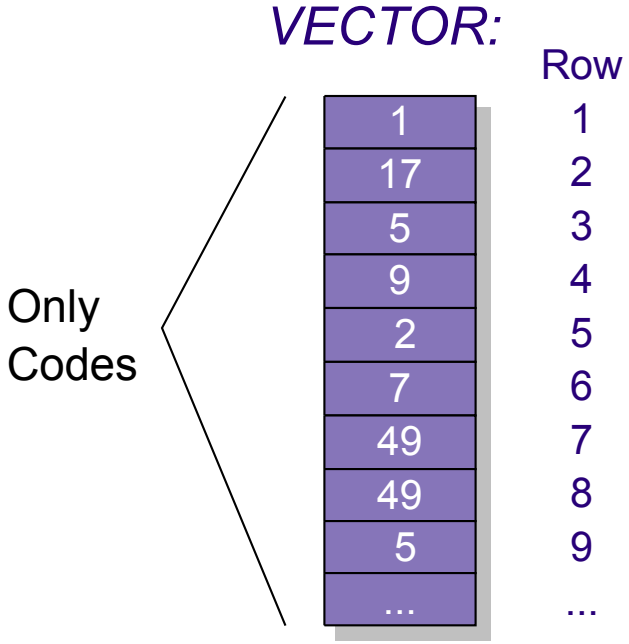
Encoded Vector Index (EVI)

Power Systems

- Object type is File, subtype is LF
- EVI is composed of two parts...

SYMBOL TABLE:

Key Value	Code	Count
Alabama	1	1000
Alaska	2	450
Arizona	3	5000
California	4	10000
Colorado	5	6500
...	...	...
Wisconsin	49	340
Wyoming	50	2760



- Symbol table contains information for each distinct key value. Each key value is assigned a unique code
  - Code is 1, 2, or 4 bytes - depending on number of distinct key values
- Rather than a bit array for each distinct key value, the index has one array of codes (a.k.a., the Vector)

Encoded Vector Indexing

Power Systems

New type of index that can significantly improve performance, especially for star schema

- 10% to 30% faster index builds
- 1/3 to 1/16 the size
- 1/2 the time for index scans
- 1/3 the time for bit map generation

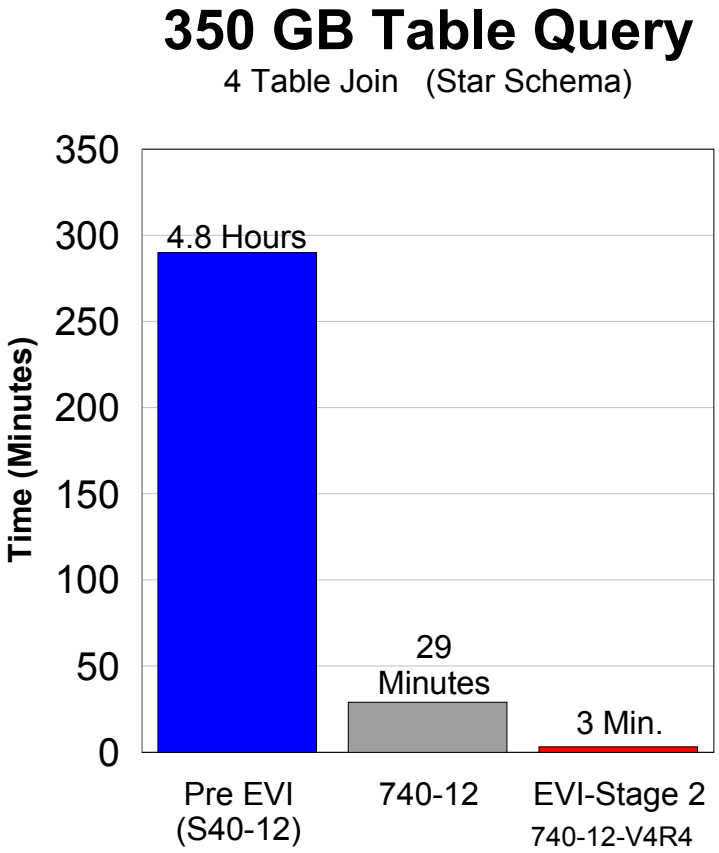
Symbol Table

Key Value	Code	First Row	Last Row	Count
Arizona	1	1	80005	5000
Arkansas	2	5	99760	7300
.....				
Virginia	37	1222	30111	340
Wyoming	38	7	83000	2760

Vector

1	13	12	28	2	17	38	2	26	33

Row 1 Row 2 ....

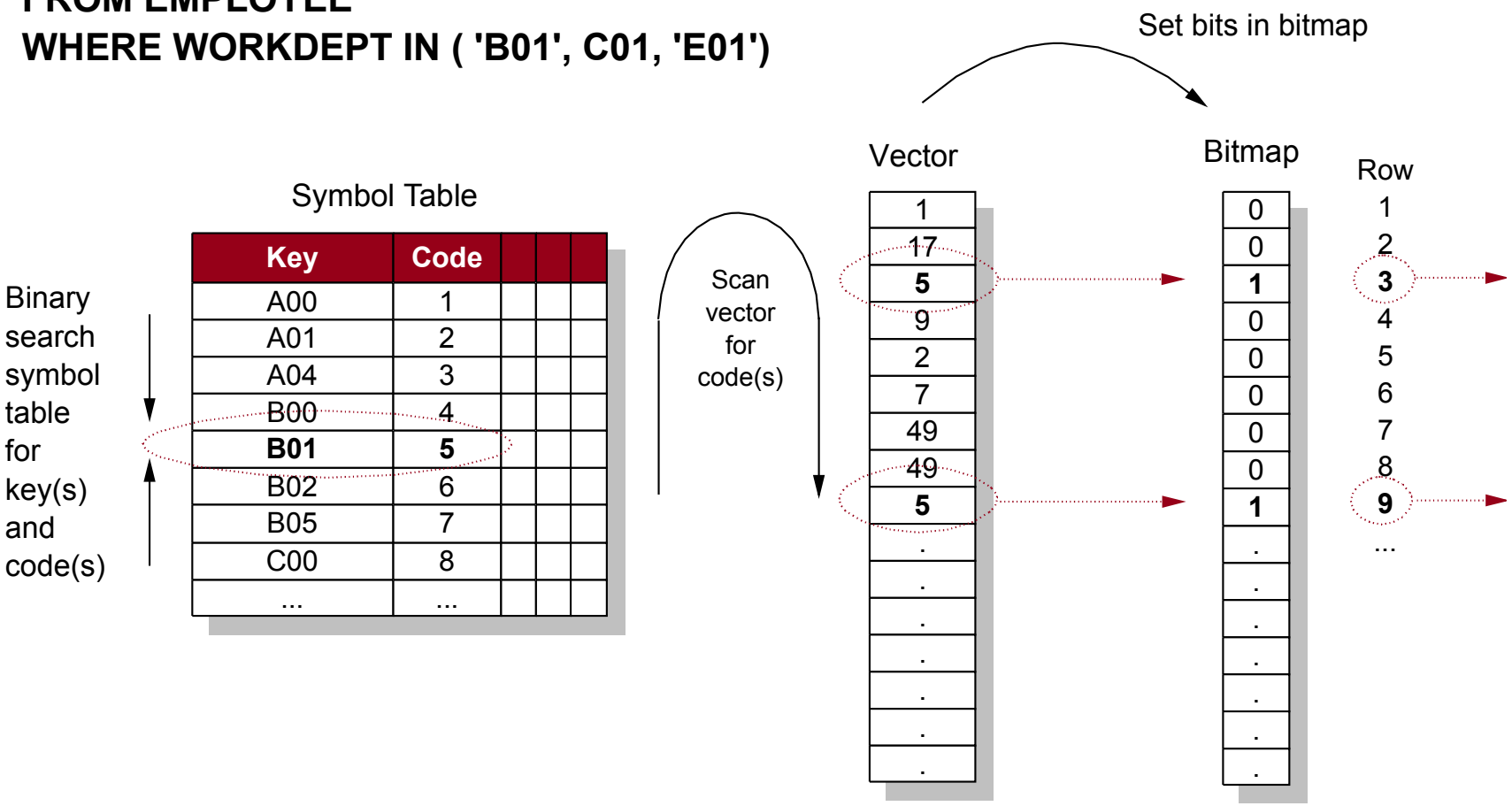


EVI Example

Power Systems

Given an EVI on table EMPLOYEE keyed on WORKDEPT...

```
SELECT *
   FROM EMPLOYEE
   WHERE WORKDEPT IN ( 'B01', C01, 'E01')
```



Indexing Strategy

- You must create some indexes
  - Statistics
  - Implementations
- Proactive
  - Create indexes over primary, foreign key columns and dependent columns
  - Create indexes for selection and joining
  - Create indexes for selection, grouping and ordering
- Reactive
  - Create indexes based on optimizer feedback
    - Visual Explain
    - Database monitors
    - Joblog messages
- Create indexes based on optimization, implementation, system resources and performance

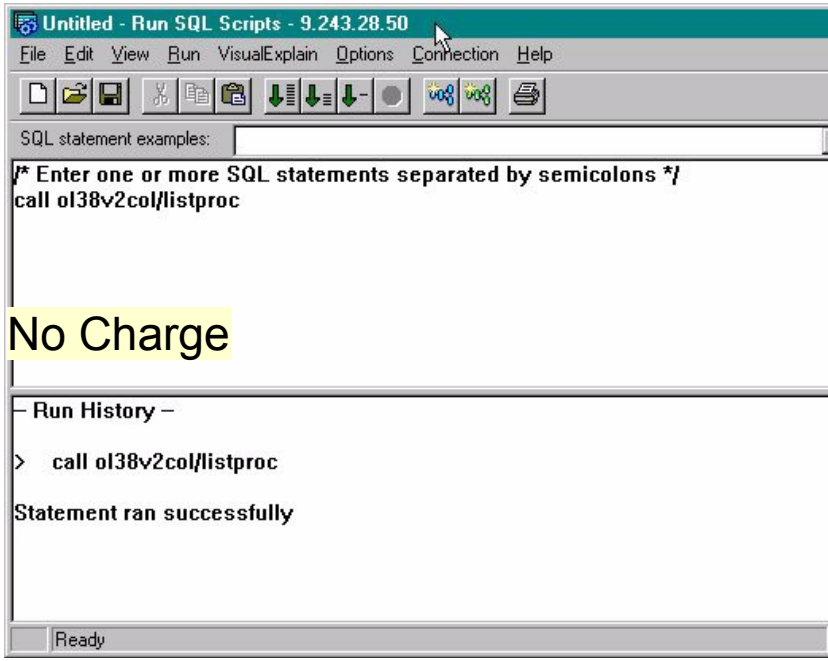
8.3 Application Development with SQL

SQL Packaging

Power Systems

- Database Manager
  - SQL parser and runtime support
  - Query Management
  - Several SQL APIs
  - Call Level Interface
  - Performance Tools
  - RUNSQLSTM

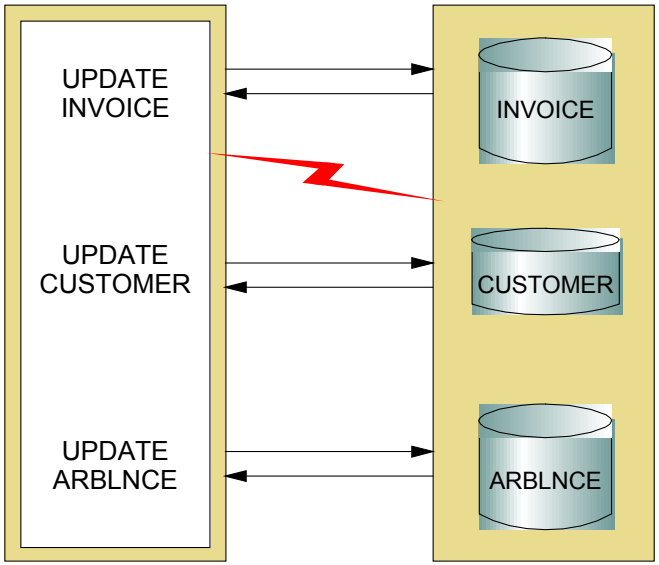
No Charge



- DB2 UDB for iSeries Query Manager and SQL Development Kit (5722-ST1)
  - STRSQL
  - SQL Preprocessors
  - Query Manager
  - SQL REXX Interface

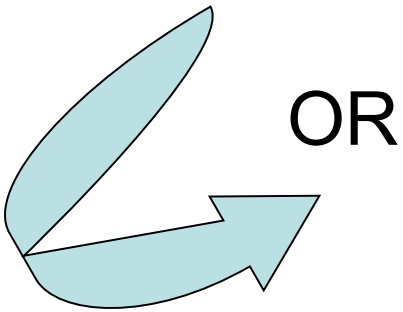
Licensed Program

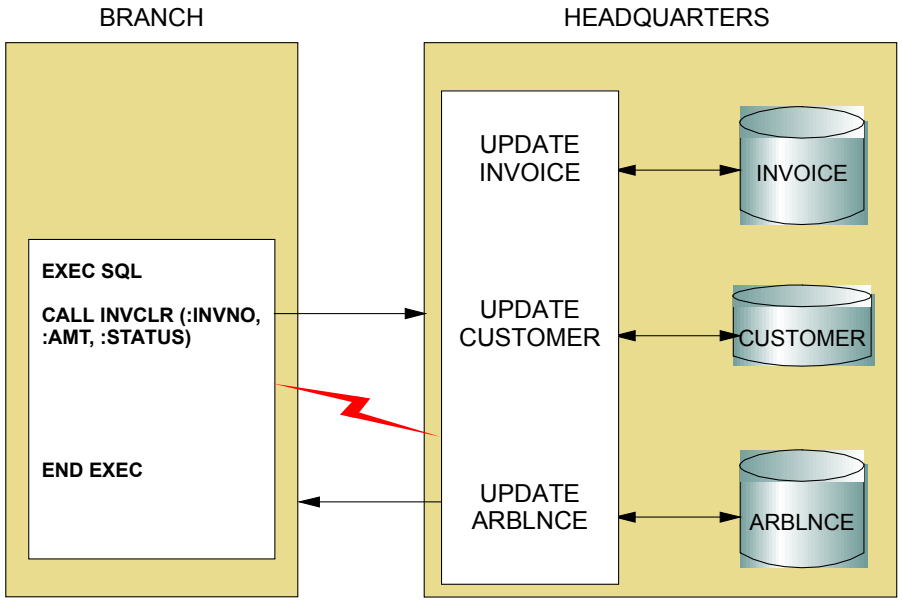
Stored Procedure



An SQL "Program" that contains:

- HLL Program with or without embedded SQL
- SQL Language
- Stored Procedure Created by CREATE PROCEDURE
- Can pass parameters
- Can return parameters, result set
- Widely used in Client / Server applications to reduce traffic





Trigger

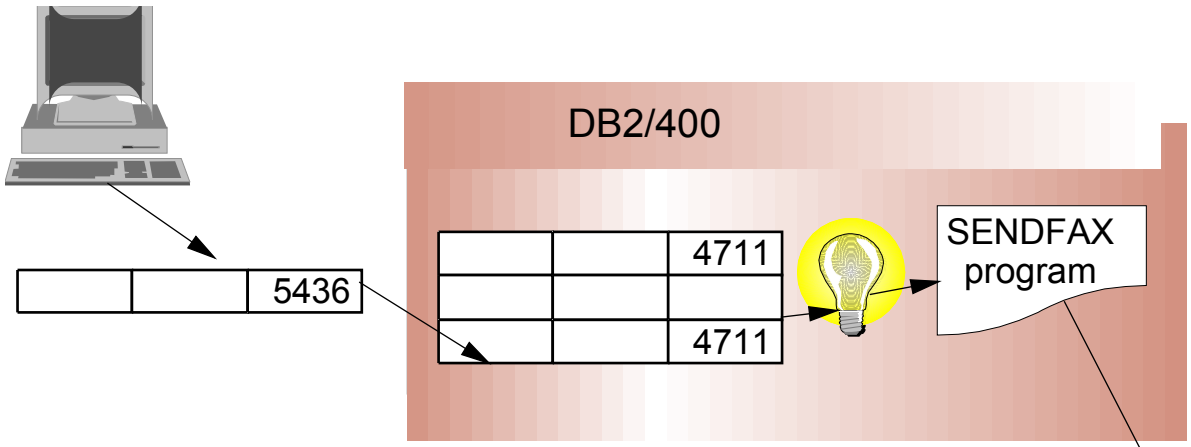
Power Systems

Application-independent
- Written once; used by many

Activated by database manager when operations performed on database
- 'Fired' by specified database operations and take action written in trigger
- Interface-independent

Triggers:
- Enforce business rules
- Enforce data validation and audit trail
- Preserve data consistency

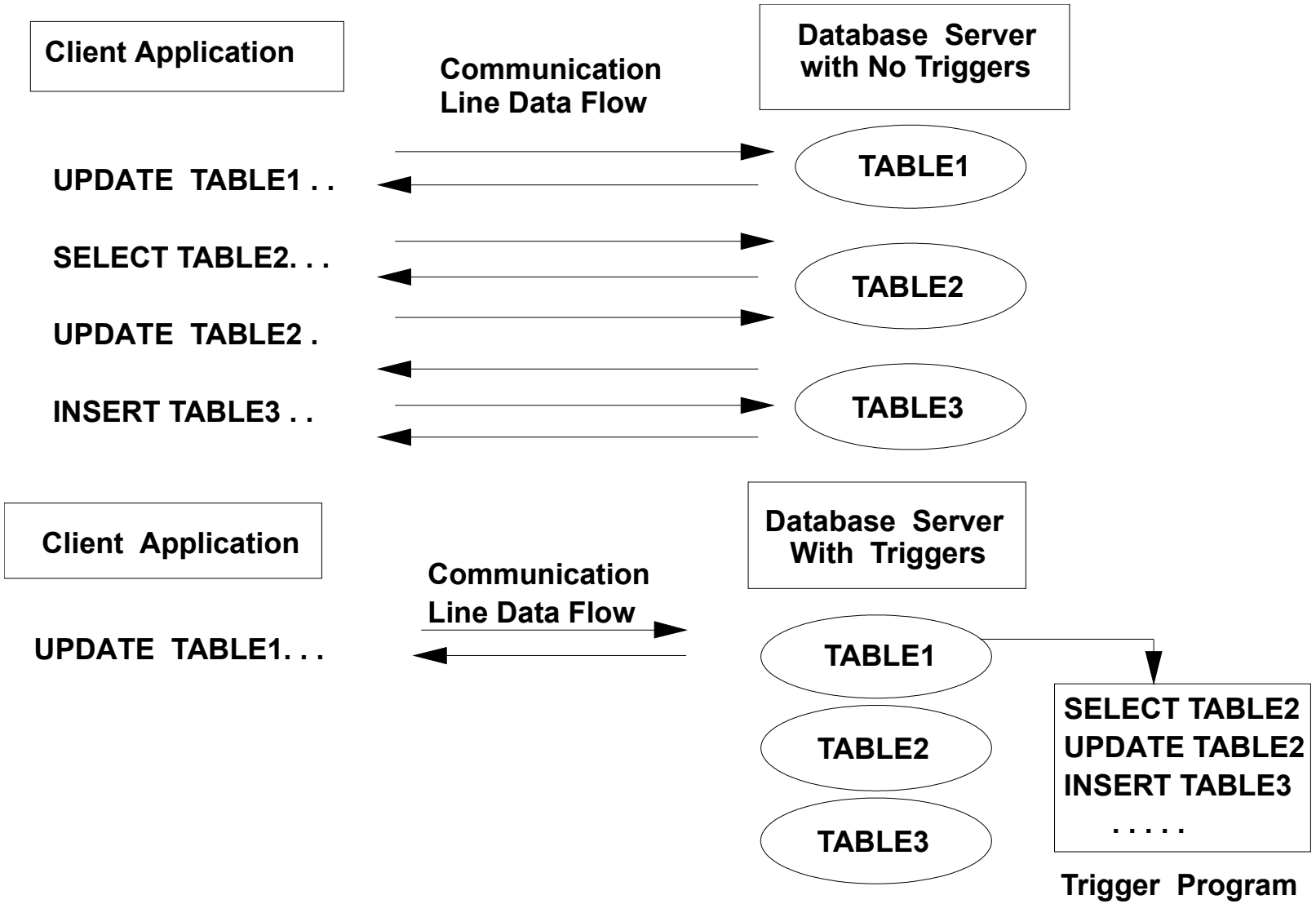
Triggers versus Stored Procedures
- Triggers invoked by database events
- Stored procedures invoked by application CALL



When a new order is inserted or updated, a trigger is fired:
- Trigger retrieves information about order and customer
- A confirmation fax is automatically sent

Trigger Example

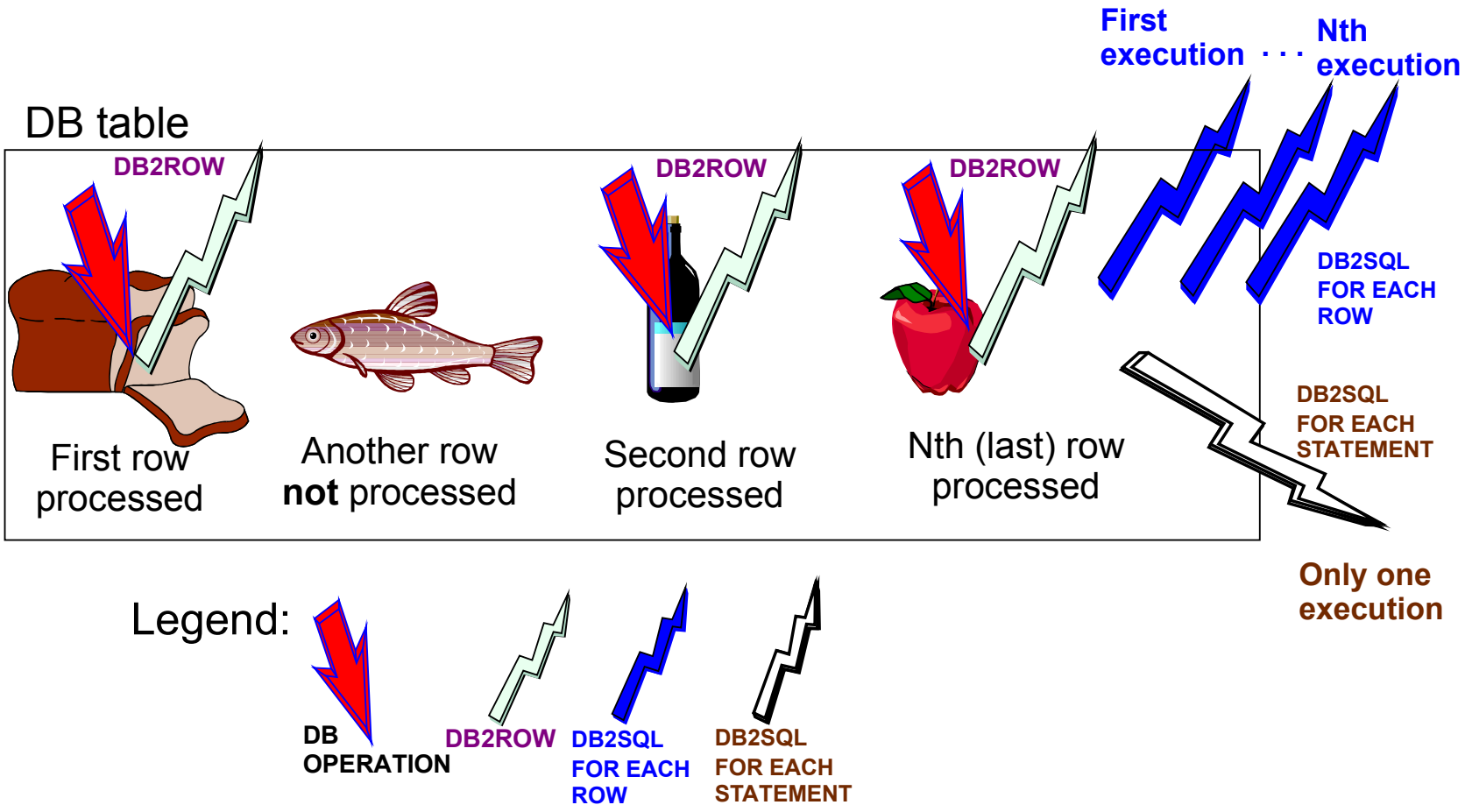
Power Systems



Trigger Mode: DB2ROW versus DB2SQL

Power Systems

- When are the different modes of trigger that are fired?



8.4 Scalability

Scalability of a DBMS

Power Systems

Capacity to meet increasing requirements

- Often caused by business factors
  - Numbers of user
  - Data volume
  - Transaction volume and complexity
- Depends on
  - Hardware
  - Operating system
  - Database configuration or design
- May not affect application design and code
  - „think big, start small“ approach

Vertical and Horizontal Scaling

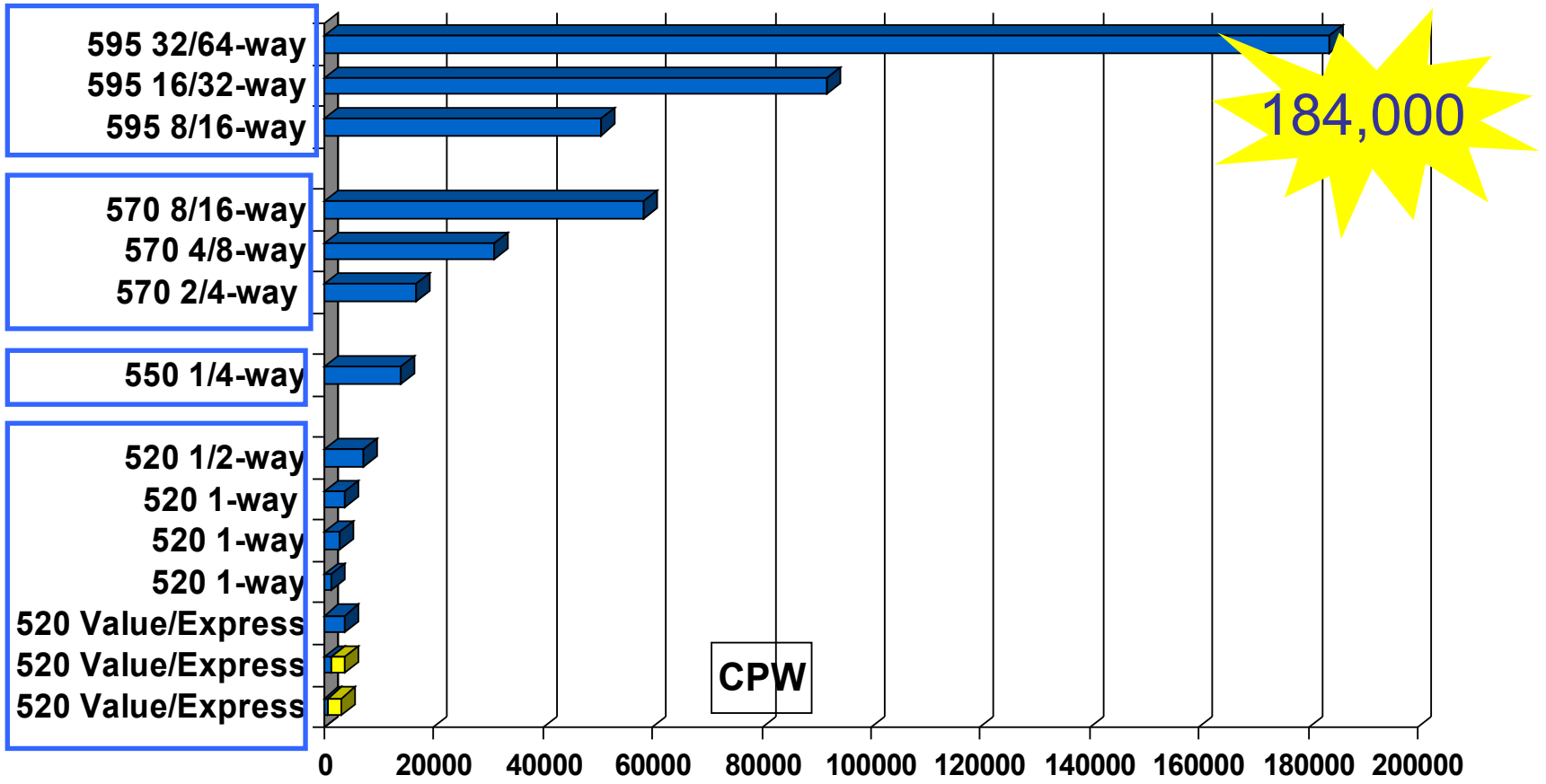
- Vertical scaling
  - typically refers to adding more processors and storage to an SMP to pump up processing capability
  - this form of scaling employs only one instance of the operating system
- Horizontal scaling
  - usually refers to tying multiple independent computers together to provide more processing power.
  - typically implies multiple instances of operating systems, residing on separate servers

9.1.1 Vertical Scaling

System i5 CPW

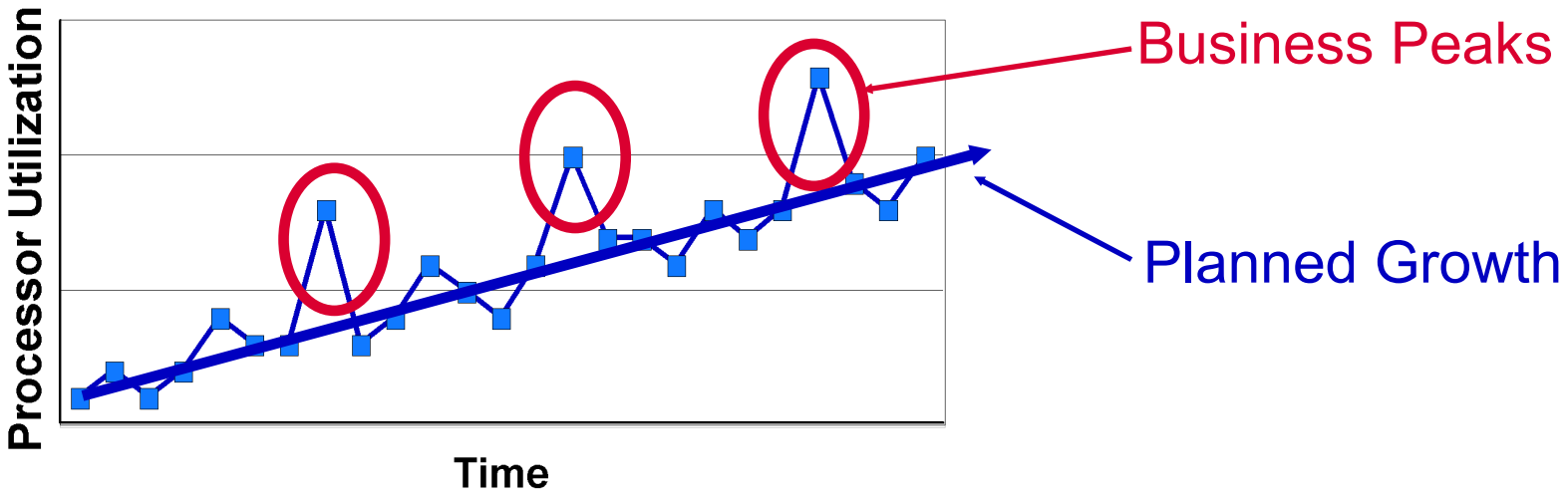
Power Systems

Note: 64-way measured as two 32-way partitions



Capacity on Demand

Power Systems



- **Permanent Capacity:** CUoD … pay when purchased (processors & memory)
- **Temporary Capacity:** On/Off CoD … pay after use (processors & memory)
- **Reserve Capacity:** CoD … pay before use (processors)
- **Trial Capacity:** CoD … no-charge for use (processors & memory)

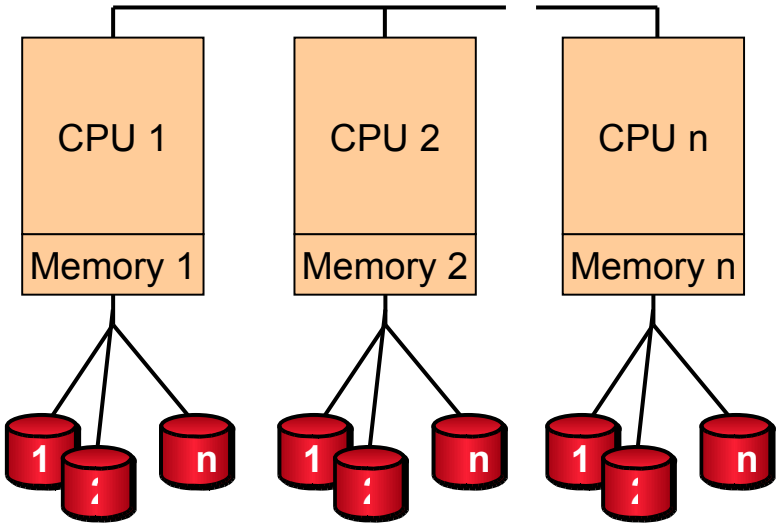
9.1.2 Database Clustering Concepts

Database Implementation Models

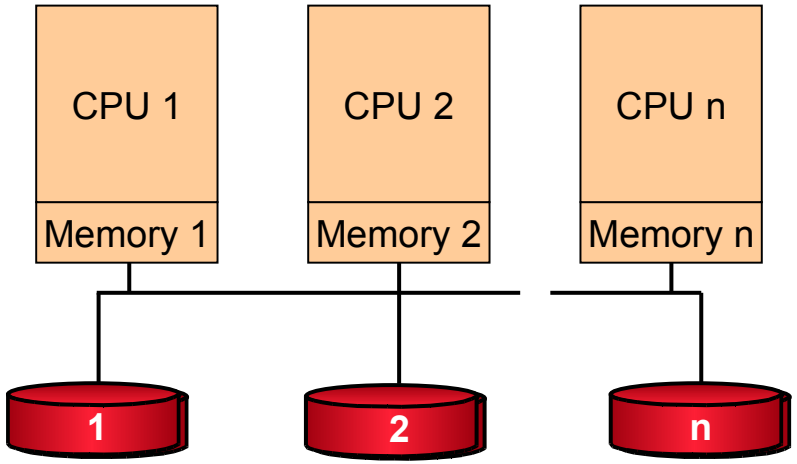
Power Systems

- Shared Nothing
  - Either a single instance SMP database using local storage or multiple database files distributed across servers that own specific data sets yet share one data dictionary
  - Multiple servers implement 'function shipping' to access specific information
  - Performance depends upon accurate data partitioning scheme
- Shared Disk
  - Cluster of SMP servers using centralized storage to preserve a single system database image with no concept of data ownership
  - All servers have equal access
  - Performance depends upon efficient inter-node synchronizations
- Shared Disk + Distributed Cache
  - Clustered server, shared disk approach with a global cache manager using memory resources of cluster nodes for avoiding disk I/O
- Federated
  - Multiple server nodes with separate databases and data dictionaries; servers do not guarantee uniqueness of records

Shared-Nothing vs. Shared Disk



- Can exploit simpler, cheaper hardware
- Almost unlimited scalability
- Work well in a high-volume, read-write environment
- Data is partitioned across the cluster



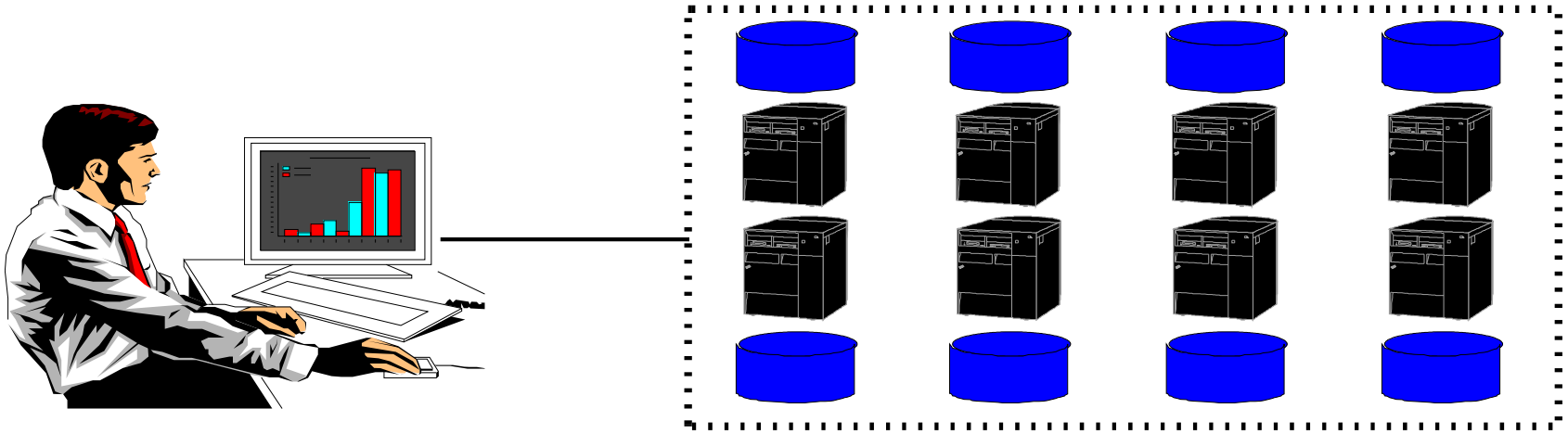
- Quick adaptability to changing workloads
- High availability
- Performs best in a heavy read environment
- Data need not to partitioned

Source:Craig S. Mullins, Database Administration – The Complete Guide to Practices and Procedures

DB2 Multisystem

Power Systems

- Loosely Coupled - Massively Parallel
  - Shared Nothing Architecture
  - Partitioned Database, Table is spread across nodes
  - User-Defined partitioning or random partitioning
- SINGLE TABLE VIEW to the application
  - Data Warehousing, OLAP, Data Mining, DSS Reporting
- Performance and Capacity Scalability virtually unlimited
- Single Table can be spread across up to 32 systems
  - up to 190 Terabytes disk space per system
- From the user's perspective, the database appears as a single database
  - the user can run queries in parallel across all the systems in the network

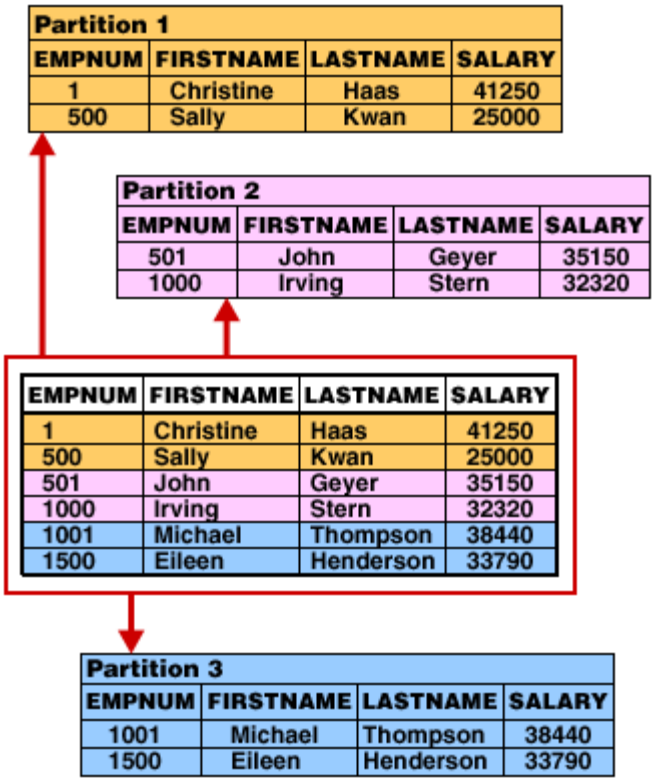


Hash Partitioning and Range Partitioning

Power Systems

Example: To partition table PAYROLL in library PRODLIB with partitioning key EMPNUM into 3 partitions, using

- Hash partitioning (places rows at random intervals)
  - CREATE TABLE PRODLIB.PAYROLL
    (EMPNUM INT, FIRSTNAME CHAR(15),
    LASTNAME CHAR(15),
    SALARY INT)
    PARTITION BY HASH(EMPNUM) INTO 3 PARTITIONS
- Range partioning
  - CREATE TABLE PRODLIB.PAYROLL
    (EMPNUM INT, FIRSTNAME CHAR(15),
    LASTNAME CHAR(15), SALARY INT)
    ´ PARTITION BY RANGE(EMPNUM)
    (STARTING FROM (MINVALUE)
    ENDING AT (500) INCLUSIVE,
    STARTING FROM (501)
    ENDING AT (1000) INCLUSIVE,
    STARTING FROM (1001)
    ENDING AT (MAXVALUE)

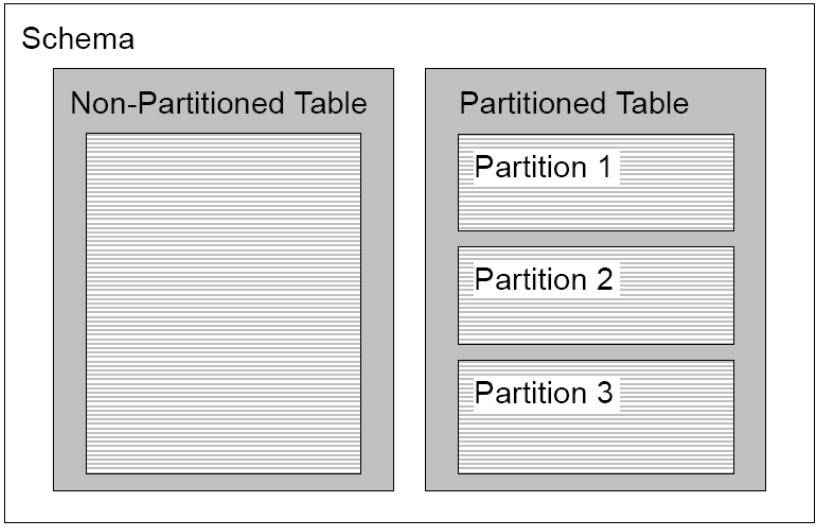


Partitioned Tables

Power Systems

- Allow a table to be stored in multiple members but treated as one table
- ONLY should be used in cases where the single table limit of 1.7 TB or 4.2 billion rows is exceeded
- Partition tables should not be used to improve performance
- Limited optimizer awareness of partitions, especially the CQE query optimizer
- Fast delete of rows in a partition is supported
- Requires the DB2 Multisystem feature of i5/OS

Partitioned Table Diagram



i5/OS does not require partitioned tables to fully exploit parallelism!

Table Sizes

To understand just how large these limits are, here is a list of various row lengths and the approximate number of rows needed to reach the 1.7TB limit:

Row Length	Number of Rows
32,766	57 million
8,192	228 million
2,048	912 million
1,024	1.8 billion
512	3.6 billion
435	4.2 billion

A partitioned table can have up to **256 partitions**,

- Each partition able to grow to the respective maximums.
- Fully populating would result in **1 trillion rows** and a **table size of 435 TB**!

Next Lecture:

- Scalability
  - SMP and Clustering
- High Availability
  - Backup and Recovery
  - Save While Active
  - Cluster
  - iASP