Replicating IMS
to
IDAA and PureData Analytics

Prepared for the:
Virtual IMS User Group

11 August 2015

Agenda

- Introduction
- DB2AA / PureData Analytics (PDA) Overview
- Update Performance Statistics
- Load & Replication Strategy
- Accelerator Only Tables
- Design Considerations
- Q & A
- Wrap Up

- **Objective** → Outline the Concepts and Methods for Applying Changes from Non-Relational Databases to DB2AA / PDA

About the Speaker

- **Scott Quillicy**
  - ✔ 30+ Years Database Experience
  - ✔ Database Software Development
  - ✔ Performance & Availability
- **Founded SQData to Provide Customers with:**
  - ✔ A Better Way of Replicating Mainframe Data...Particularly IMS
  - ✔ Solutions that Combine Expertise with Technology
  - ✔ Technology Built Around Best Practices
- **Specialization**
  - ✔ Data Replication
  - ✔ IMS to Relational
  - ✔ Heterogeneous Database Integration
  - ✔ Continuous Availability
  - ✔ Advanced Data Analytics

About SQData

- **Enterprise Class Changed Data Capture & Replication**
- **Core Competencies**
  - ✔ High-Performance Changed Data Capture (CDC)
  - ✔ Non-Relational Data → IMS, VSAM, Flat Files
  - ✔ Relational Databases → DB2, Oracle, SQL Server, etc.
  - ✔ Deployment of Complex Data Integration Solutions
  - ✔ Continuous Availability of Critical Applications
  - ✔ Data Conversions / Migrations

- **Customer Use Cases**
  - ✔ Near-Real-Time Operational Data Stores (ODS) from Multiple Sources
  - ✔ Continuous Availability → Active/Active, Active/Passive
  - ✔ ETL (Bulk Data Extracts/Loads)
  - ✔ Application Integration
  - ✔ Business Event Publishing
  - ✔ Data Warehouse Population

Why Replicate IMS to DB2AA / PDA?

- Provide a Method of Analyzing Data Outside of IMS
- Real-Time Business Intelligence / Advanced Analytics
- Bulk Loads can be Resource Intensive
- Combine with Data from other Applications
- Save Significant CPU Cycles for Intense Queries
- Compliment Established Data Warehouse(s)

PureData Analytics (PDA)

- Netezza Appliance
- Acquired by IBM in 2010 – Data Analytics Strategy

Transforms the User Experience

- ✓ Purpose-built analytics engine
- ✓ Integrated database, server and storage
- ✓ Standard interfaces
- ✓ Low total cost of ownership

Speed: 10-100x faster than traditional systems

Simplicity: Minimal administration and tuning

Scalability: Peta-scale user data capacity

Smart: High-performance advanced analytics

DB2 Analytics Accelerator (DB2AA)

- Netezza Appliance Coupled to DB2
- Minimal Application Changes Required

The hybrid computing platform on zEnterprise

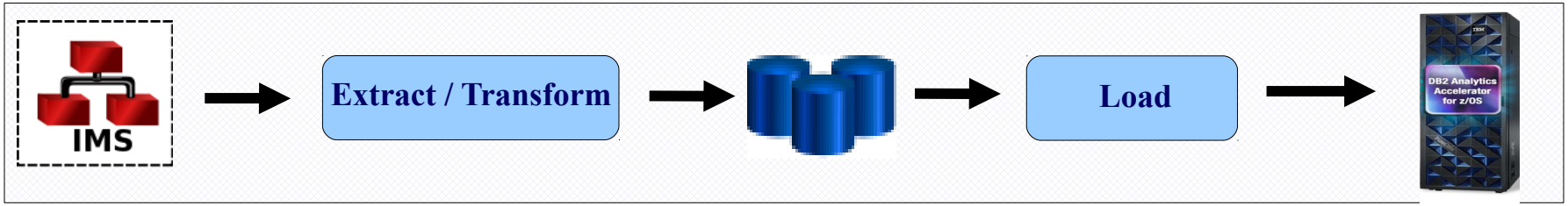
- *Supports transaction processing and analytics workloads concurrently, efficiently and cost-effectively*
- *Delivers industry leading performance for mixed workloads*

DB2 Analytics Accelerator and DB2 for z/OS

The Role of ETL and CDC

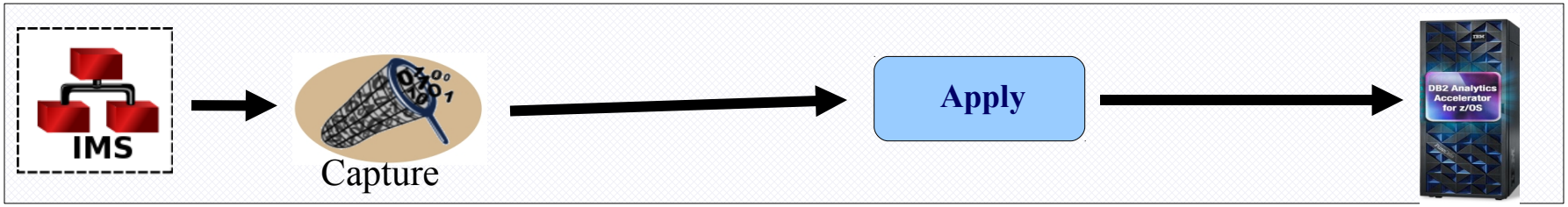
ETL (Extract, Transform, Load):

- ✓ Full Data Extract / Load
- ✓ Data Transformation Logic Defined in this Step
- ✓ Iterative Process – Must be Fast and Efficient
- ✓ Should Minimize Data Landing



CDC (Changed Data Capture):

- ✓ Move Only Data that has Changed
- ✓ Re-Use Data Transformation Logic from ETL
- ✓ Near-Real-Time / Deferred Latency



Performing the Initial Load

- Transformation / Mapping Logic Done Here → Reused in CDC
- Should be Able to Run Against Live Sources
- Make Sure to Truncate Before Loading → Otherwise Duplicates
- May be Used in Leu of Incremental Updates (CDC)
- **NZLOAD**
  - ✔ Native Loader
  - ✔ Allows for Rapid Loading of PDA Tables
- **IDAA Loader (recommended component)**
  - ✔ IBM Product Offering
  - ✔ Allows Simultaneous Loading into DB2 and IDAA
  - ✔ Allows IDAA Only and AOT Loads

Insert, Update and Delete Behavior

- **Updates → Delete / Insert Pairs**

- **Inserts**
  - Appends Data to End of File → Very Fast
  - Speed is Based on Number of Rows Being Inserted

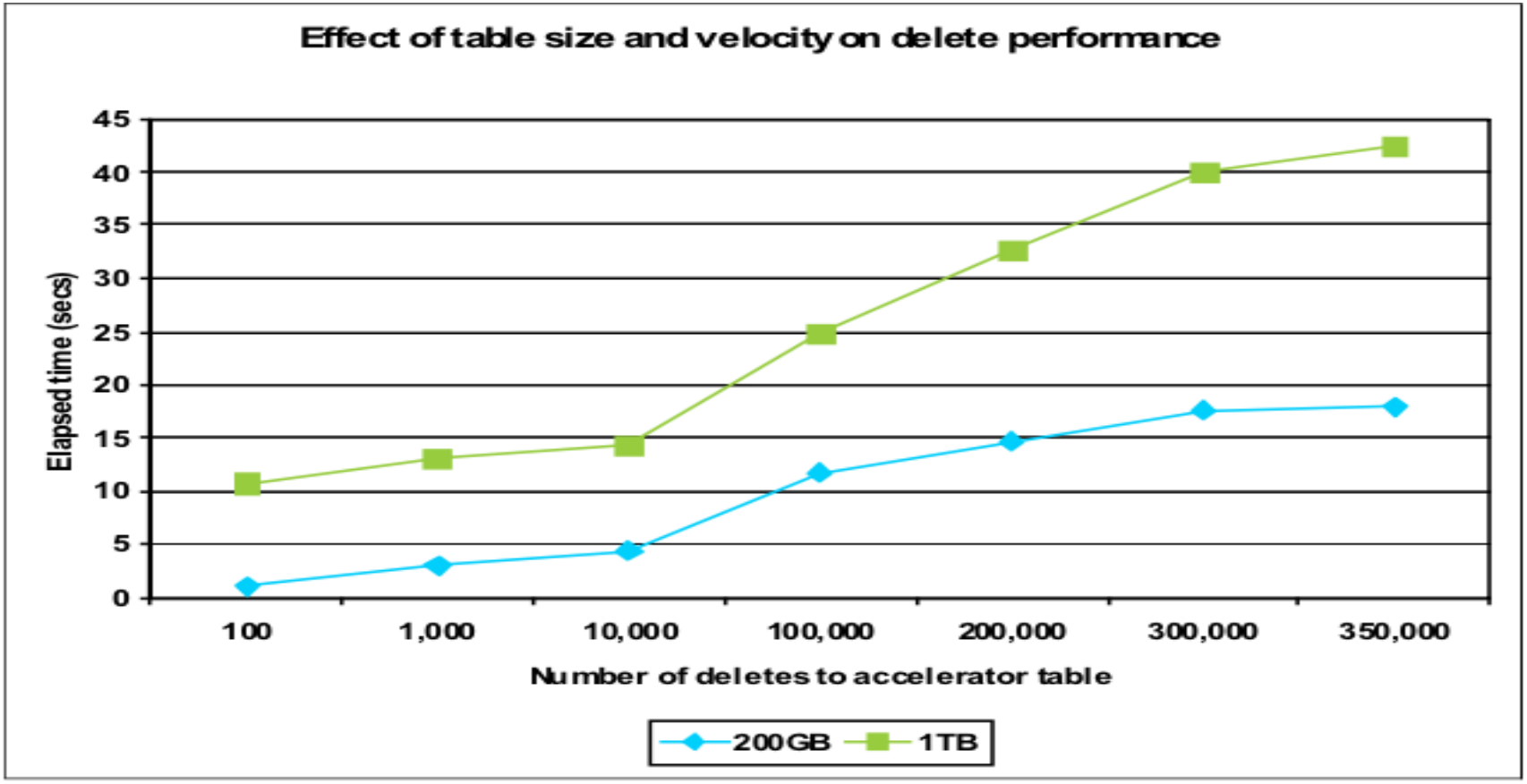
- **Deletes**
  - Must Scan Entire File
  - Select Row – Update with Delete Flag
  - Replace Row in Place
  - Groom Process Cleans up Files
  - Speed is Based on Size of File and Number of Rows Being Deleted

Insert / Delete Performance

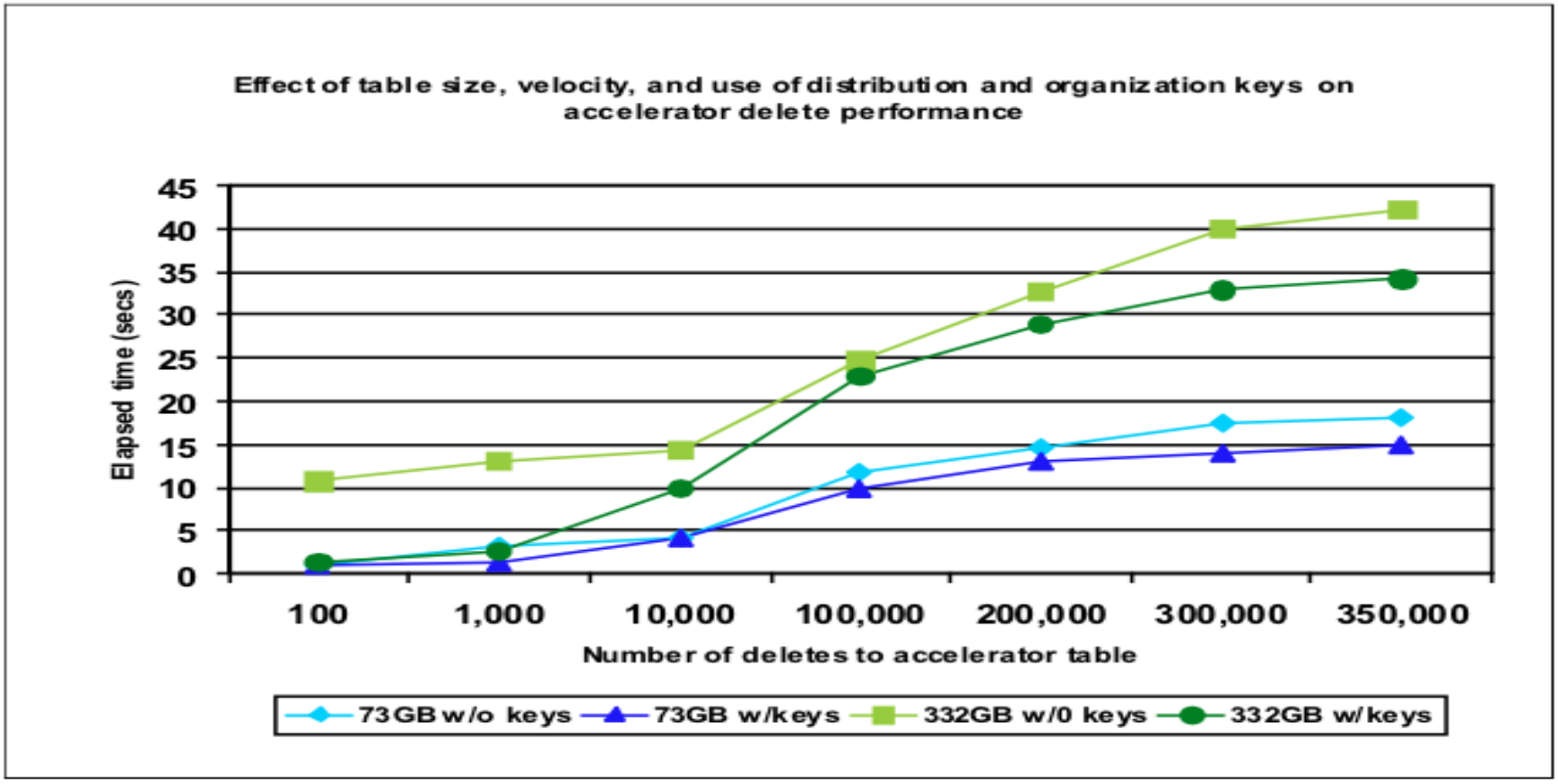
- With and Without Distribution and Organizing Keys

	Rows		Elapsed time (seconds)			
Test scenario	**Inserted**	**Deleted**	**INSERT**	**DELETE**	**Total**	**Improvement**
Separate **INSERT** and **DELETE** with random distribution	43,916,377	22,007,406	7093	5997	13,091	
Separate **INSERT** and **DELETE** with **DISTRIBUTE** and **ORGANIZE** on unique key	43,916,377	22,007,406	5200	5088	10,288	27%
Mixed **INSERT** and **DELETE** with random distribution	43,916,377	22,007,406	9232	5733	14,966	
Mixed **INSERT** and **DELETE** with **DISTRIBUTE** and **ORGANIZE** on unique key	43,916,377	22,007,406	6544	4448	10,992	36%

Delete Performance → Table Size



Delete Performance → Table Size and Keys



Replication Strategy

- **General Approach:** Apply in 'Batches' vs Continuous
- **Expectations on Latency** → Minutes vs Sub-Second
- **Time Series Data**
  - ✔ Tracks Lineage of Changes
  - ✔ Common Deployment to Track
    - Customer Tendencies
    - Campaign Effectiveness
    - Correlation of Events
  - ✔ All Changes Applied as Inserts
  - ✔ **Options** → Batches or Continuous Feed
- **Synchronized Data**
  - ✔ Source and Target Match at any Given Time
  - ✔ Inserts, Updates and Deletes must be Processed
  - ✔ **Option** → Batches

Target Key Selection

- **Golden Rule** → Good Distribution = Good Performance
- **Objective**
  - ✔ All Tables are Distributed Across All Active Database Blades
  - ✔ All Queries Run Parallel Against All Active Database Blades
  - ✔ All Loads Run Parallel Against All Active Database Blades
- **Distribution Key Selection**
  - ✔ Primary Key of Source Data
  - ✔ Columns Used for Joins
  - ✔ Columns with High Cardinality
  - ✔ Columns Frequently Aggregated On
- **Organizing Key Selection**
  - ✔ Only Use on Tables with > 1M Rows
  - ✔ One or More Columns of Primary Key → Incremental Update Performance
  - ✔ Columns Used as Common Predicates

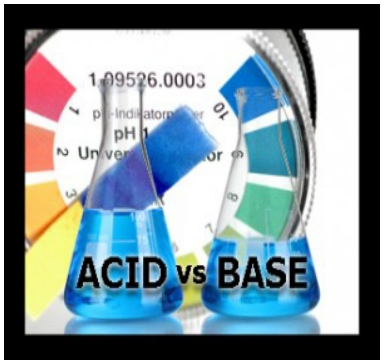
ACID vs BASE

- **ACID** → Properties Guarantee DB Transactions are Processed Reliably
  - **A**tomicity → All or Nothing...either the Transaction Commits or it Doesn't
  - **C**onsistency → Transaction brings DB from One Valid State to Another
  - **I**solation → Concurrency
  - **D**urability → Once a Transaction Commits, it Remains Committed

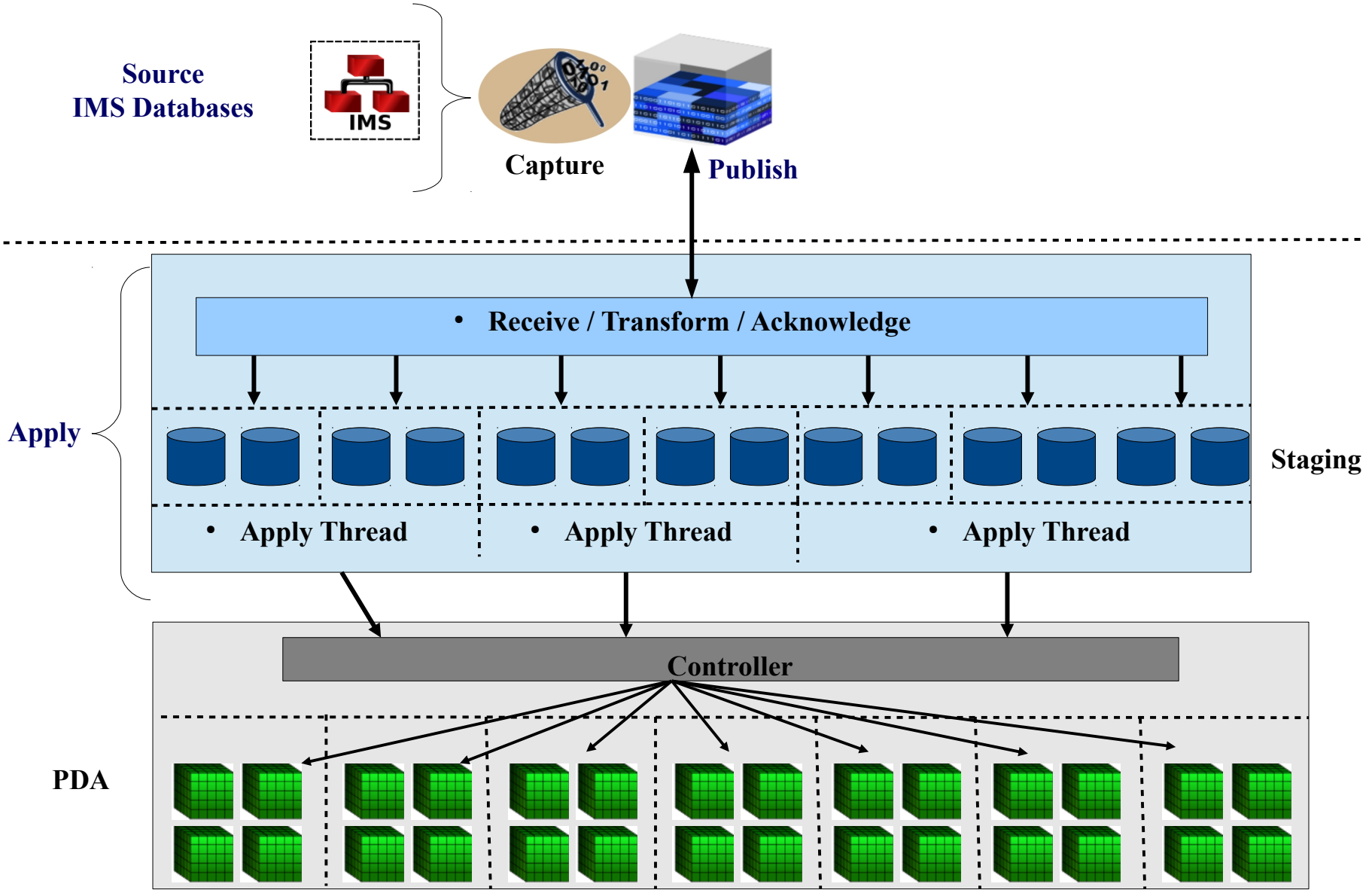
- **BASE** → Eventual Consistency
  - **B**asically **A**vailable → Data is There...No Guarantees on Consistency
  - **S**oft State → Data Changing Over Time...May Not Reflect Commit Scope
  - **E**ventual Consistency → Data will *Eventually* become Consistent

More Info: Charles Rowe – Shifting pH of Database Transaction Processing

http://www.dataversity.net/acid-vs-base-the-shifting-ph-of-database-transaction-processing/



IMS to PDA Replication Illustration



IMS to DB2AA Replication

- **Accelerator Must Know About Apply Processes**
- **Recommend: PTF 5**
- **Accelerator Only Tables**
  - ✔ Allows Update DML against Tables in Accelerator
  - ✔ Apply Process can Perform Inserts/Deletes via DB2
  - ✔ Decent Throughput Today → Will Only Get Better in the Future

- **AOT Restrictions**
  - ✔ Currently only Supported in DB2 V10
  - ✔ Single Row Inserts – Multi-Row Inserts in Development
  - ✔ Transient in Nature
  - ✔ Cannot be Enabled for Incremental Update
  - ✔ Cannot Backup/Recover via Utilities

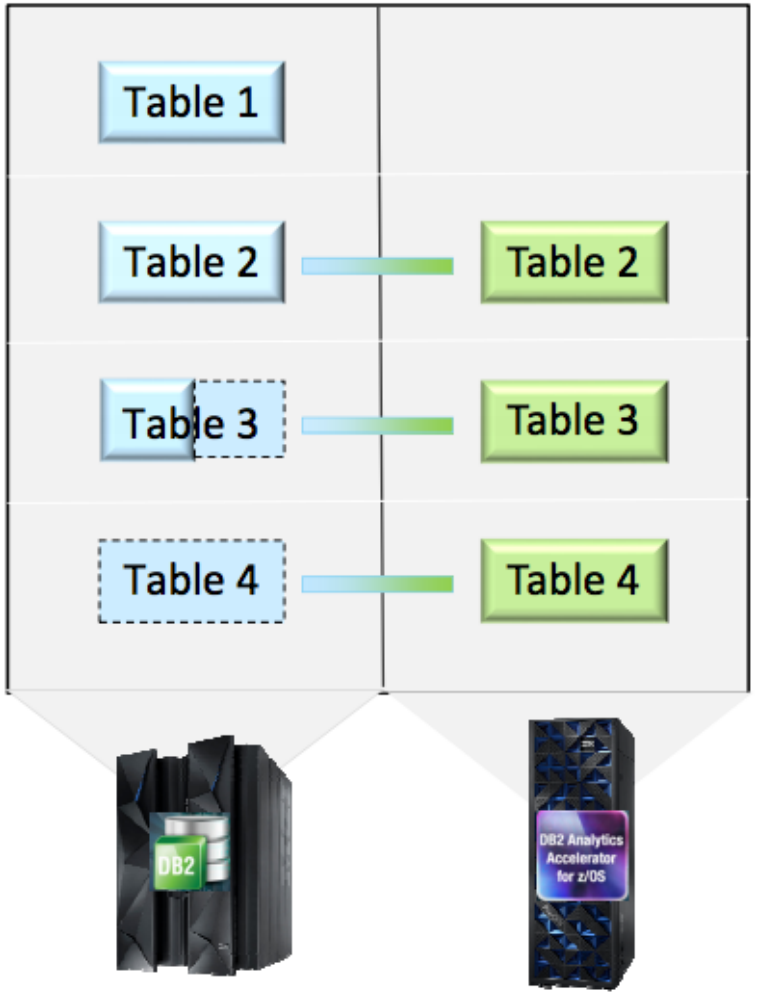
DB2AA Tables

Non-accelerated DB2 table
- Data in DB2 only

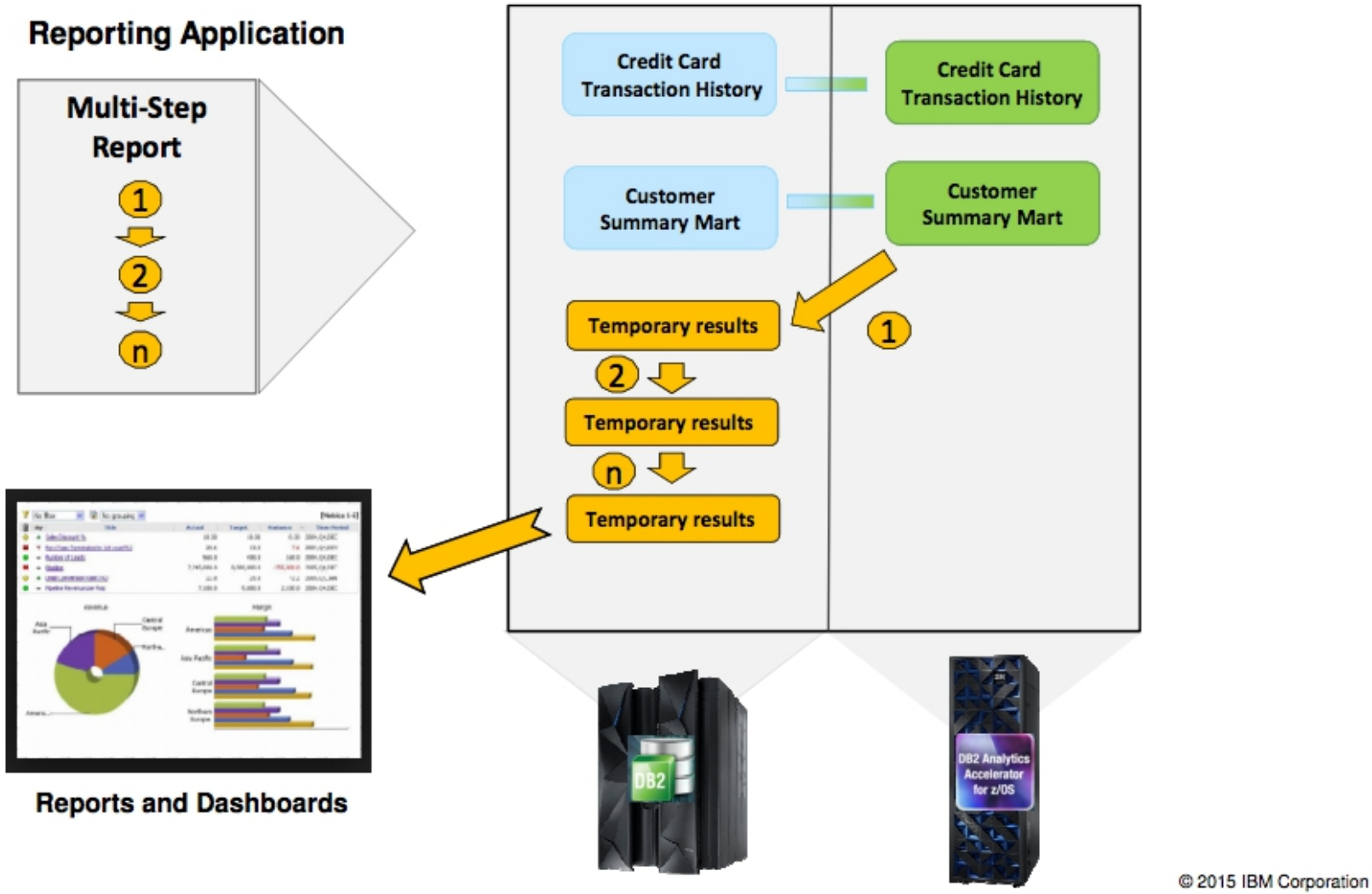
Accelerated DB2 table
- Data in DB2 and the accelerator

Archive table / partition
- Empty read-only partition in DB2
- Partition data is in accelerator only

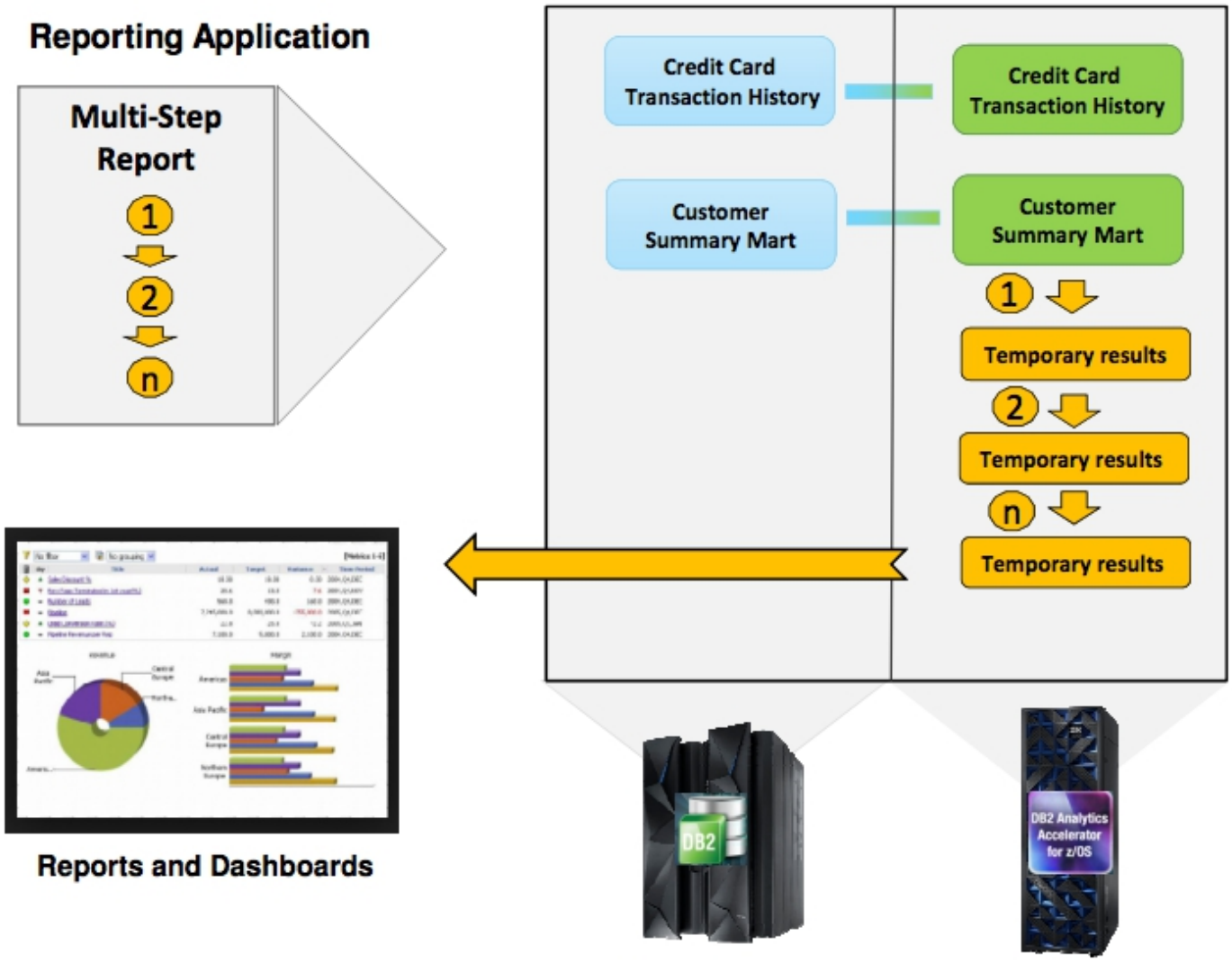
Accelerator-Only table (AOT)
- "Proxy table" in DB2
- Data is in accelerator only



Query Workload Before AOTs



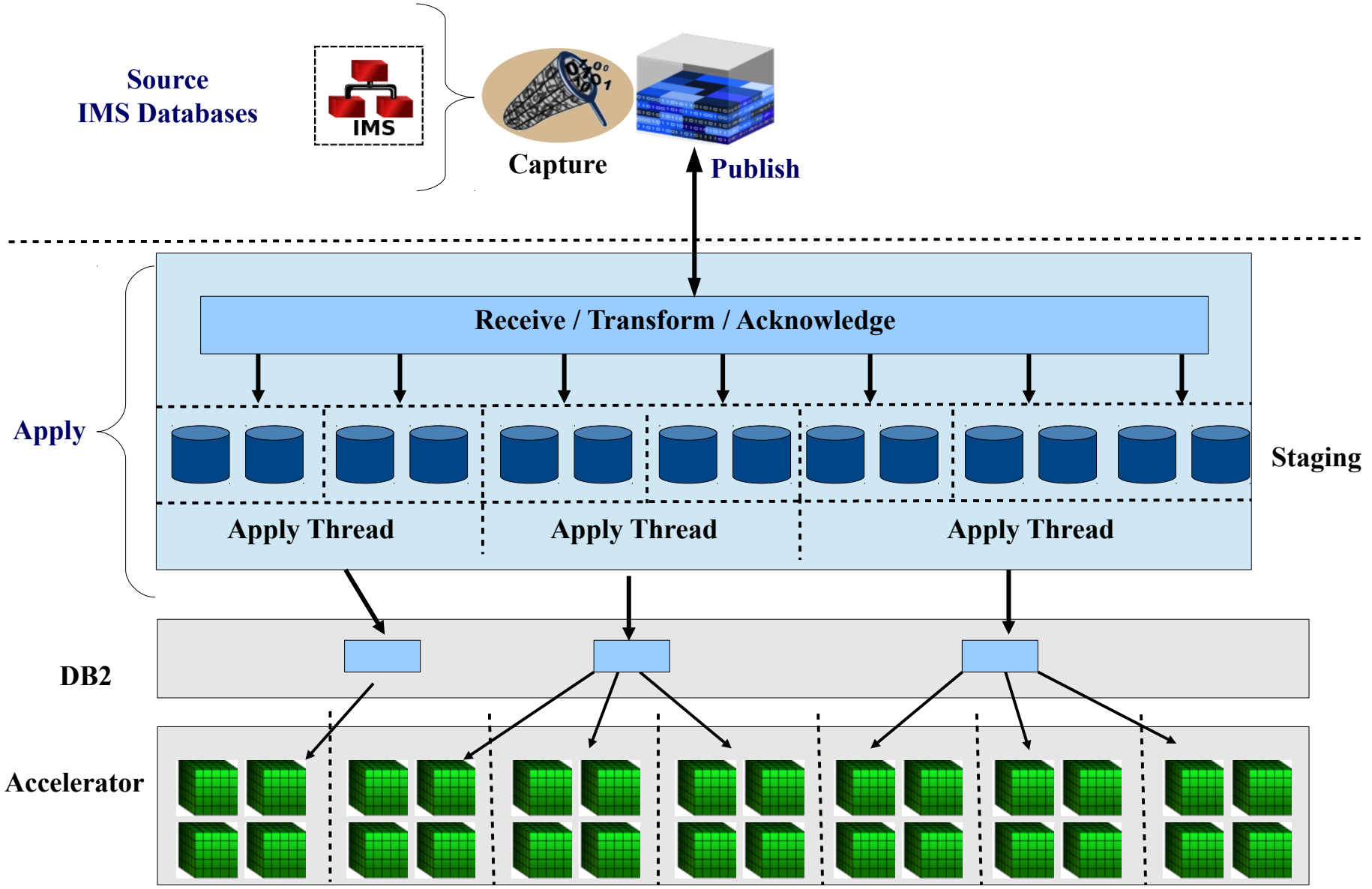
Query Workload → After AOTs



Running Queries on the Accelerator

- **Special Register** → CURRENT QUERY ACCELERATION
- **Mandatory Criteria**
  - ✔ All Tables in Query (Join) Exist in the Accelerator
  - ✔ Query is Defined as Read Only – Except AOTs
  - ✔ Cursor is Not Defined as Scrollable
  - ✔ Row Set Cursor is Not Remote
  - ✔ Query is from a Package and Not a Plan with DBRMs
  - ✔ Query is a SELECT or INSERT FROM SELECT Statement
- **SQL Restrictions**
  - ✔ Special Registers Other Than:
    - CURRENT DATE
    - CURRENT TIME
    - CURRENT TIMESTAMP
  - ✔ Sequence Expressions → NEXTVAL or PREVVAL
  - ✔ User Defined Functions (UDFs)
  - ✔ MIN / MAX with Strings or More than Four (4) Arguments

IMS to DB2AA Replication Illustration



Design Considerations

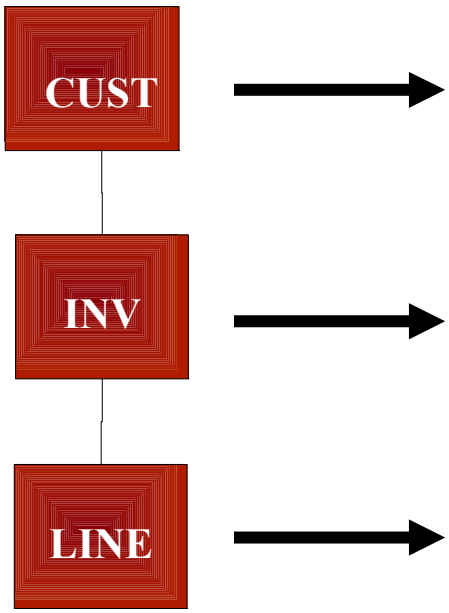
- **The MOST IMPORTANT Component**
- **IMS to Relational Model is a Good Start**
  - ✔ Design for End Queries
  - ✔ *Some* Denormalization → 2nd Normal Form
- **Load and CDC**
  - ✔ Define Transformation/Business Logic Once
  - ✔ Stream Loads → Minimize Data Landing
  - ✔ CDC → Determine Latency Requirements and Related Tables (for apply groups)
  - ✔ Transaction Volume and Apply Method
- **Recovery**
  - ✔ Same as for Traditional Relational
  - ✔ Stream Loads...CDC Catchup
- **Monitoring**
  - ✔ System Log Monitoring
  - ✔ On-Demand via a Dashboard
  - ✔ Notification of Key Events

Common IMS Data Challenges

- **Code Page Translation**
- **Invalid Data**
  - Non-Numeric Data in Numeric Fields
  - Binary Zeros in Packed Fields (or Any Field)
  - Invalid Data in Character Fields
- **Dates**
  - Must be Decoded / Validated if Target Column is DATE or TIMESTAMP
  - May Require Knowledge of Y2K Implementation
  - Allow Extra Time for Date Intensive Applications
- **Repeating Groups**
  - Sparse Arrays
  - Number of Elements
  - Will Probably be De-normalized
- **Redefines**
- **Binary / 'Special' Fields**
  - Common in Older Applications Developed in 1970s / 80s
  - Generally Requires Application Specific Translation

General Approach

- ✓ Each Segment Maps to One (1) or More Tables
- ✓ Helpful → Keep Source Fields and Target Column Names Similar
- ✓ Design Considerations
  - Duration → Lower for Rehost...Higher for BI/DW
  - Strong Target Data Types will Require Additional Transformation
  - Be Careful to Avoid the 'Over Design'
- ✓ **Best Practice**: Keep Things as Simple as Possible



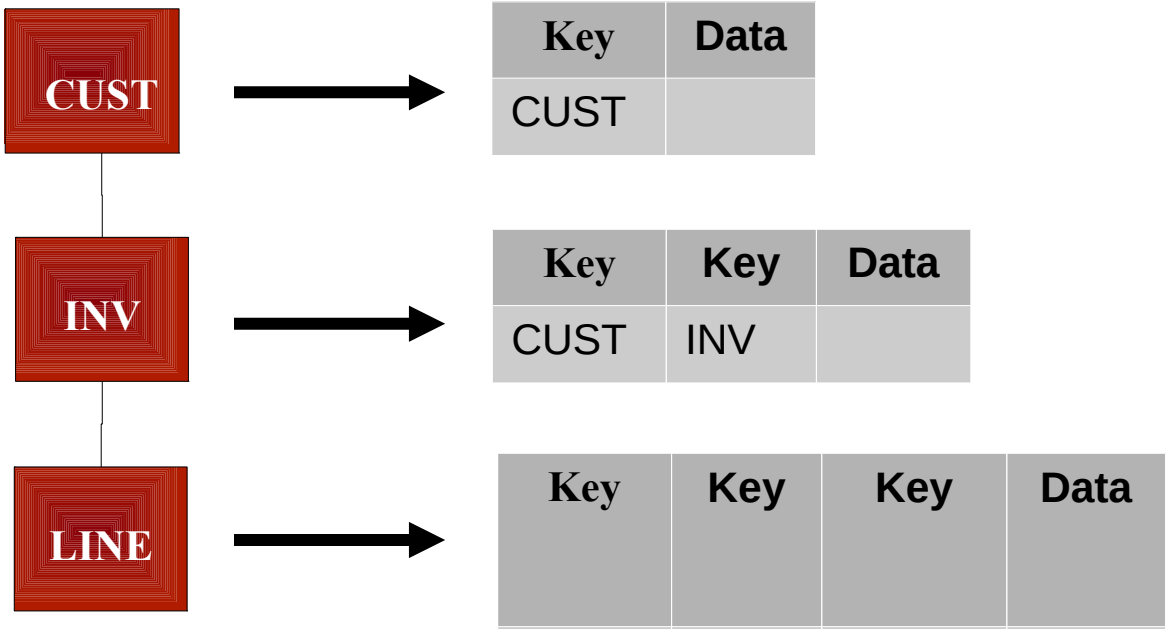
Key	Data
CUST	

Key	Key	Data
CUST	INV	

Key	Key	KeyA	KeyB	Data
CUST	INV	LINE-A	LINE-B	

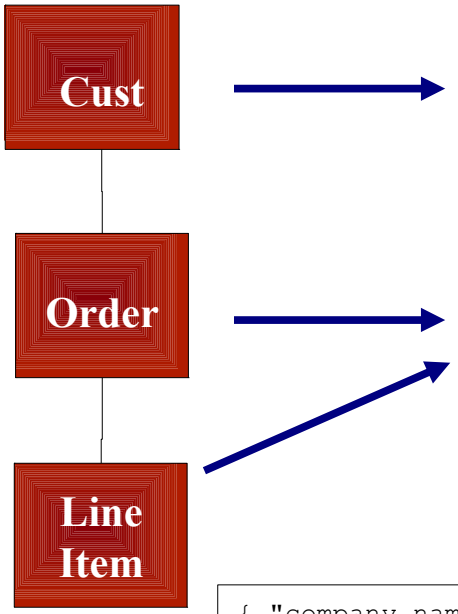
IMS to Traditional Relational Model

- Normalized → at Least 2nd Normal Form
- Each Segment Typically Maps to One (1) or More Tables



Alternative → IMS to Big Data Model

- De-Normalized / Minimal Normalization
- Degree of Data Redundancy → Trade-Off for Query Performance



Key	Data
Cust#	

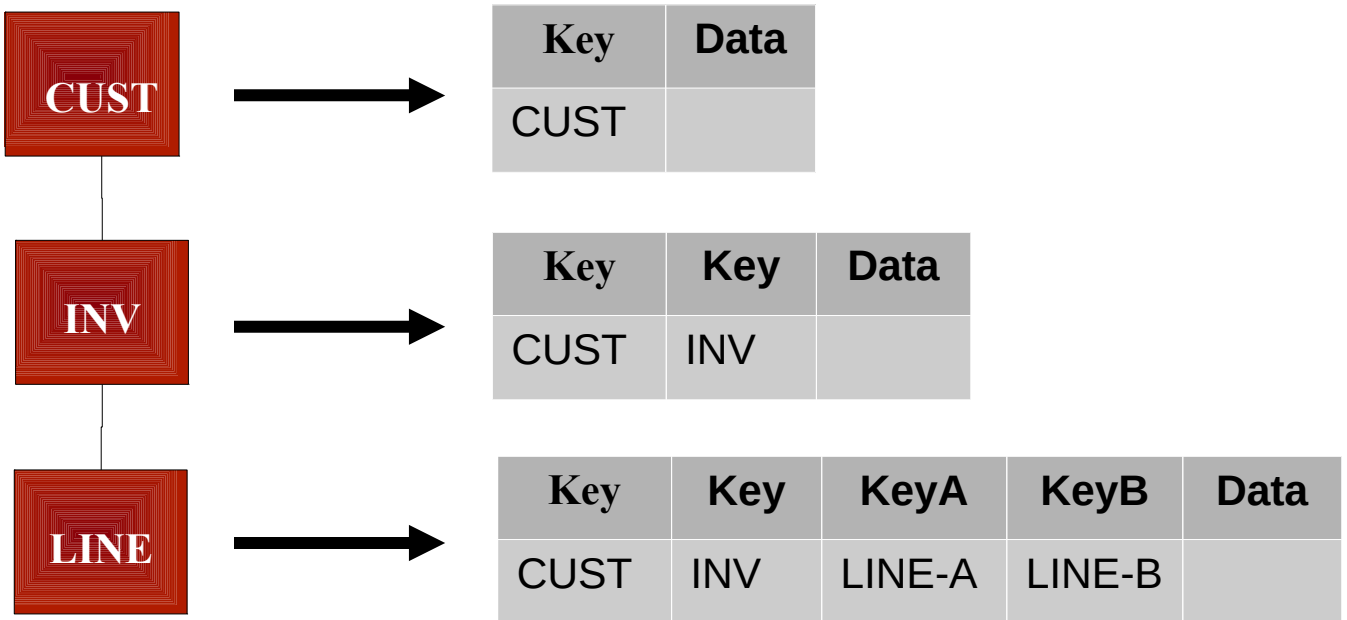
Key	Data	Data	Data	Data	Data	Data
Order#	Cust#		Line #		Line#	

```
{ "company_name" : "Acme",
  "cust_no"      : "20223",
  "contact" :{ "name" : "Jane Smith",
               "address" : "123 Maple Street",
               "city" : "Pretendville",
               "state" : "NY",
               "zip"   : "12345" }
}
```

```
{ "order_no" : "12345",
  "cust_no"  : "20223",
  "price"  : 23.95,
  "Lines" : { "item" : "Widget1",
              "qty"  : "6",
              "cost" : "2.43"
              "item  : "Widge2y"
              "qty"  : "1",
              "cost" : "9.37"
            },
}
```

Keys

- ✓ Fairly Straightforward → IMS Key Structure Simplifies Things
- ✓ Carry Parent Keys in Dependent Tables
- ✓ Use these Unique Keys as Distribution Keys in Accelerator
- ✓ Plan on Source Keys Containing Multiple Fields with Different Data Types
  - Character, Packed, Binary



Redefined Fields

- ✓ Extends Analysis Timeline More Often than Not
- ✓ Requires Consult with SME and/or Research to Determine Which Field to Use
- ✓ Options for Simple Redefines:
  - Map Least Restrictive Field (PIC X)
  - Map Both Fields

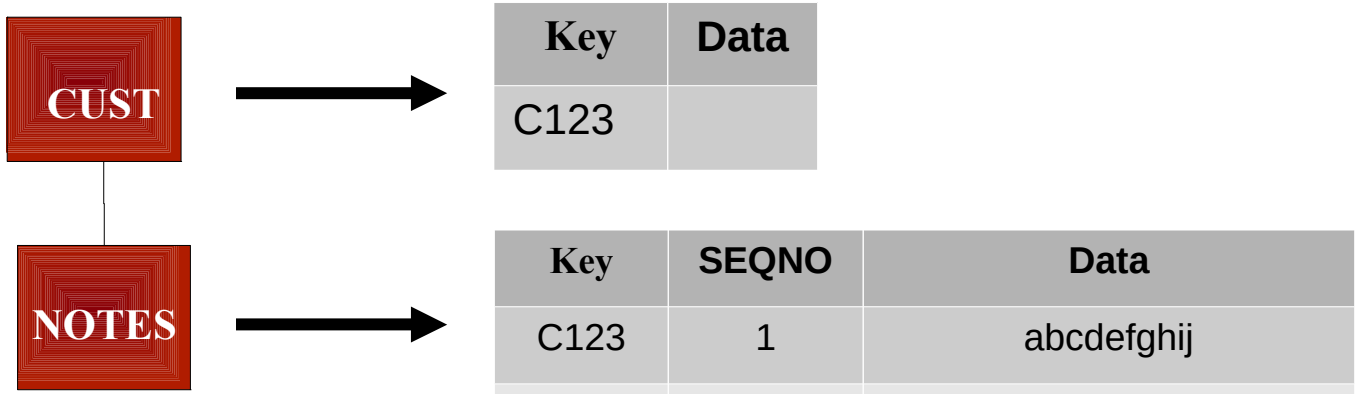
```
05   ACCOUNT-ID                                PIC 9(7).
05   ACCOUNT-ID REDEFINES ACCOUNT-NO           PIC X(7).
```

- ✓ Options for Complex Redefines:
  - Map More Granular Field(s) → Will Require More Data Cleansing / Transformation
  - Map All Fields

```
05   ACCOUNT-ID                                PIC X(5).
05   ACCOUNT-ID REDEFINES ACCOUNT-NO.
       10  ACCOUNT-PREFIX              PIC X(1).
       10 ACCOUNT-NUMBER              PIC S9(7) COMP-3.
```

Non-Keyed Segments

- ✓ Commonly Used for Text / Comments
- ✓ Straightforward for ETL
  - Unload in Order of Occurrence
  - Optional: Use a Sequence Number to Keep Things in Order on Target Side
- ✓ Tricky for CDC
  - Only Have Access to Parent Key(s)
  - Option 1: Set Apply Key to Include All Non-Keyed Data (exclude sequence #)
  - Option 2: Fully Materialize All Non-Keyed Segments when 1 Changes
  - Make Sure Your ETL/CDC Tool Can Handle Non-Keyed Segments



Key	Data
C123	

Key	SEQNO	Data
C123	1	abcdefghij
C123	2	klmnopqrs

Repeating Groups: Relational

- ✓ Typical Candidates for Normalization Based on # Occurs
- ✓ Options:
  - Low # Occurs → Keep in Same Table as Rest of Segment
  - Map to Separate Table – Requires a Sequence Number

```
05  ACCT-ID                          PIC 9(7).
05  ACCT-CRDATE                      PIC X(8).
05  ACCT-BALANCE                     PIC S9(13)V99 COMP-3.
05  ACCT-ACTIVITY OCCURS 100 TIMES.
      10 ACT-DATE                    PIC 9(8).
      10 ACT-TYPE                    PIC X.
      10 ACT-AMOUNT                  PIC S9(11)V99  COMP-3.
```

ACCT_ID	ACCT_CRDATE	ACCT_BALANCE
12345	20120617	9000.00

ACCT_ID	SEQNO	DATE	TYPE	AMOUNT
12345	1	20120618	D	8000.00
12345	2	20120622	D	1000.00

Alternative → Repeating Groups: Big Data

- ✓ All Occurrences into the Same Target
- ✓ No Need for Sequence Number

```
05  ACCT-ID                              PIC 9(7).
05  ACCT-CRDATE                          PIC X(8).
05  ACCT-BALANCE                        PIC S9(13)V99 COMP-3.
05  ACCT-ACTIVITY OCCURS 100 TIMES.
      10 ACT-DATE                       PIC 9(8).
      10 ACT-TYPE                       PIC X.
      10 ACT-AMOUNT                     PIC S9(11)V99  COMP-3.
```

ACCT_ID	ACCT_CRDATE	BALANCE	DATE	TYPE	AMOUNT	DATE	TYPE	AMOUNT
12345	20120617	9000.00	20120618	D	8000.00	20120622	D	1000.00

Summary

- PDA / DB2AA are High Value Data Analytics Weapons
- Analytics against Current Data is ***Critical***
  - ✔ Maintaining a Competitive Edge
  - ✔ Real-Time Trend Detection
  - ✔ Customer Tendencies and Correlation with other Events
- Design Carefully
  - ✔ Keep as Simple as Possible
  - ✔ Start with Traditional IMS to Relational Model
  - ✔ Ask for Assistance from Those with the Experience
- Stay Flexible
  - ✔ Technology is Changing Quickly
  - ✔ Solution Must be Portable to Another Platform (i.e. Spark, Hadoop, etc.) with Minimal Changes

Replicating IMS
to
IDAA and PureData Analytics

Prepared for the:
Virtual IMS User Group

11 August 2015