IBM Software Group

EII - ETL - EAI
What, Why, and How!

Information
on demand
Are you building a
foundation for the future?

Tom Wu 巫介唐, wuct@tw.ibm.com
Information Integrator Advocate
Software Group
IBM Taiwan

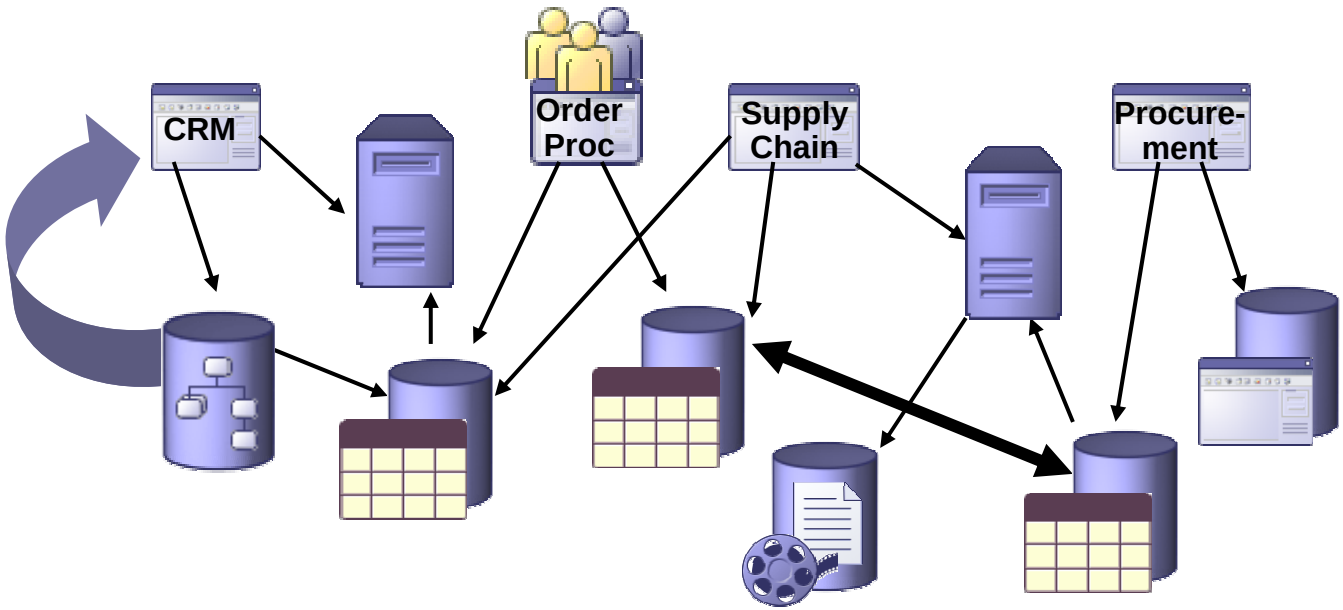
ON DEMAND BUSINESS

Agenda

- **Data Integration Challenges and IBM Vision**
- Definitions and Patterns
- Data Integration Approaches
  - ETL vs. EII vs. EAI

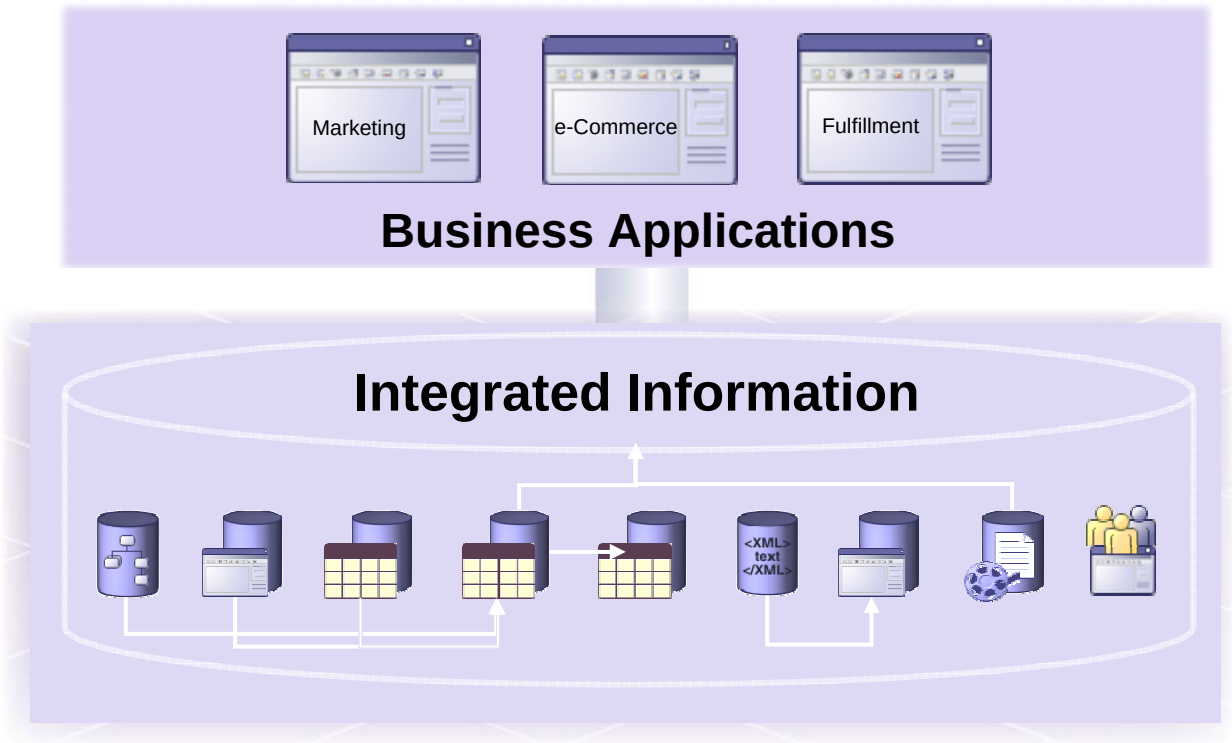
Today's World: Complex and Costly

- Heterogeneous, distributed data
- Inconsistent islands of information underlie applications
- Complications from M&A and departmental purchases
- Complex & costly copy synchronization
- Inconsistent and poor quality data
- No feedback on quality of service
- Impossible to support business transformation



IBM Information Integration Vision

Delivering accurate, consistent, timely, and coherent business information



Get the right information, in the form you want, whenever you need it.
Isolate applications from information complexity.

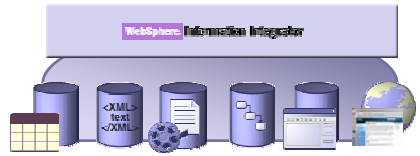
Integration Challenges

- Lack of confidence in the correctness of information
- Lack of interface design principles and common formats
- Definition of correct format and semantic layer to be able to merge two or more disparate repositories/applications data
- Definition of an Integration Governance Model
- Creation of a methodology to document technical items such as record definitions, structures, interfaces, and flows across the whole organization
- Defining rules and methods to maintain consistency across all data integration projects

Agenda

- Data Integration Challenges and IBM Vision
- **Definitions and Patterns**
- Data Integration Approaches
  - ETL vs. EII vs. EAI

EII - Enterprise Information Integration



Optimized & transparent data access and transformation layer providing a single relational interface across all enterprise data

- It enables the integration of structured and unstructured data
  - to provide real-time read and write access,
  - to transform data for business analysis and data interchange, and
  - to manage data placement for performance, currency, and availability.

EAI - Enterprise Application Integration

Message-based, transaction-oriented, point-to-point (or point-to-hub) brokering and transformation for application-to-application integration

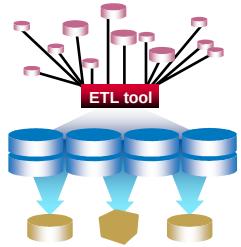
- The core benefits offered by Enterprise Application Integration are:
  - A focus on integrating both business-level processes and data
  - A focus on reuse and distribution of business processes and data
  - A focus on simplifying application integration by reducing the amount of detailed, application specific knowledge required by the users

EAI Types

- There are four types of Enterprise Application Integration:
  - User Interface Level
  - Method Level
  - Application Interface Level
  - Data Level – Many organizations choose Data Level EAI as a starting point

Middleware and EAI leverage message brokers, application servers, distributed objects, and distributed agents. Transactional Middleware is becoming a more popular solution that relies on method sharing to provide a reliable messaging framework.

ETL - Extract - Transform - Load



Set-oriented point-in-time transformation for migration, consolidation, and data warehousing.

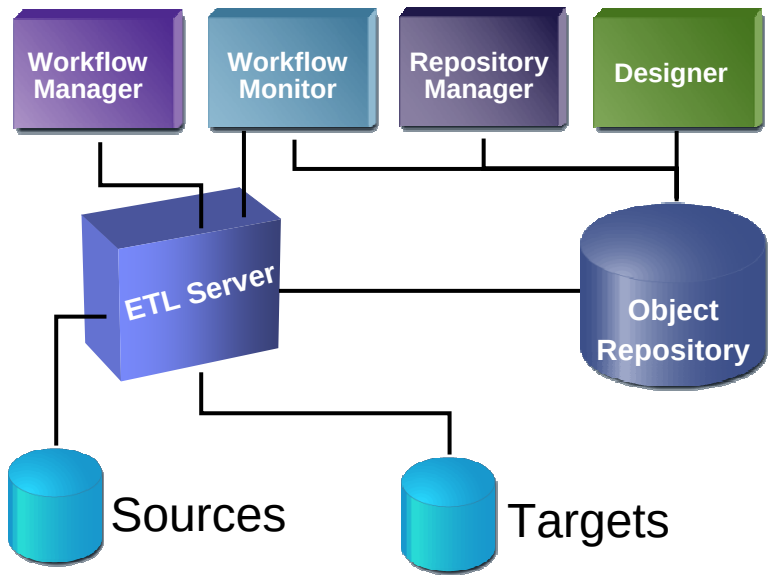
- Designed to process very large amounts of data, ETL provides a suitable platform for:
  - Improved productivity by reuse of objects and transformations
  - Strict methodology
  - Better Metadata support, including impact analysis

Role of ETL tools

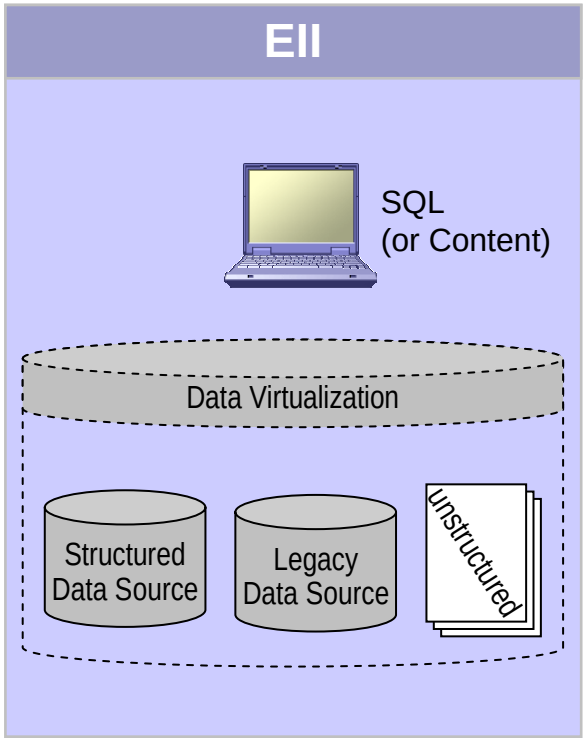
- Scheduling
- Parallel and concurrent workloads
- “Message” based designs
- Impact analysis
- Metadata gathering and management

ETL Architecture

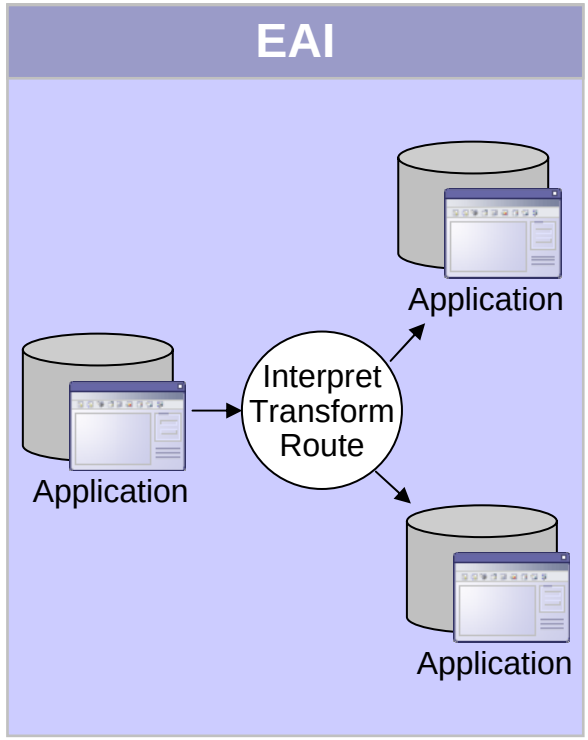
- Based on metadata repositories and ETL engines
- Parallel ETL engines offer performance and scalability
- Schedule or Event Driven
- Workflow engine – Integration with business processes



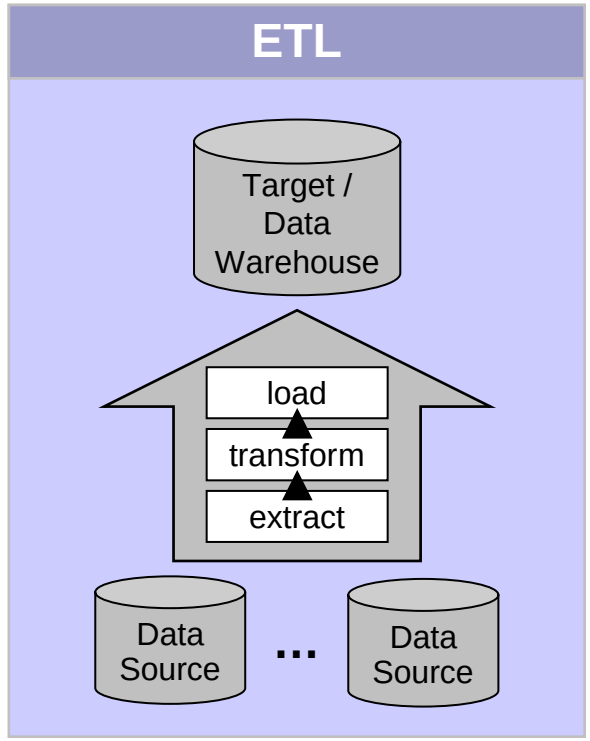
Information Integration – Data Patterns



- Real-time information access
- Federation of data from multiple sources
- Dynamic drill down
- Semi-structured & unstructured data



- Process based integration of application data
- Message-based, transaction-oriented processing
- Workflow and data orchestration, content-based routing

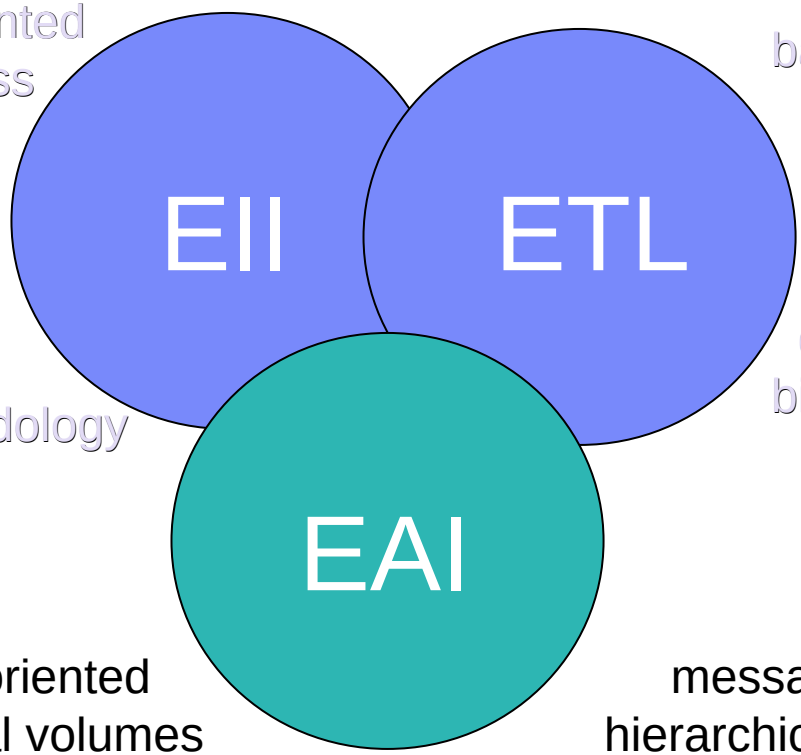


- Bulk data integration
- Set-based & hierarchical transformations
- High scale, batch-oriented data delivery

Unique Characteristics

query oriented
dynamic, real-time access
transparency across
different sources
metadata support

less a tool than a SQL
access layer and methodology



batch oriented
latency tolerant
complex transformations
massive volumes
code-less specification
deep metadata support
bi-tool integration

transaction-oriented
transactional volumes
developer audiences

C, Java, etc.

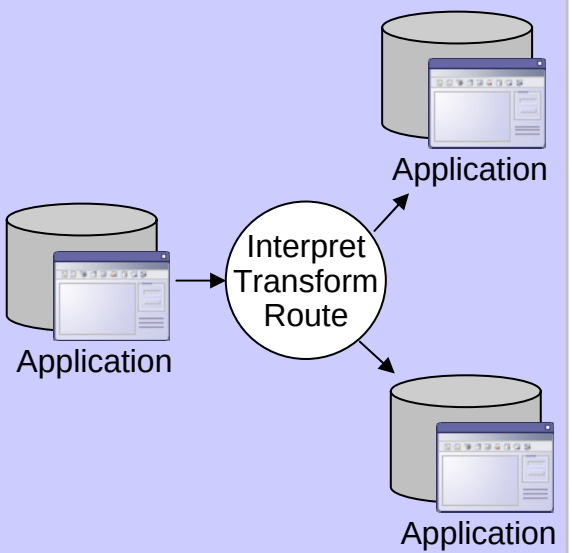
message-based
hierarchical structures
EDI, SWIFT, HIPPA

lower emphasis on metadata integration

ensured delivery 24 by 7 security encryption restart/recovery

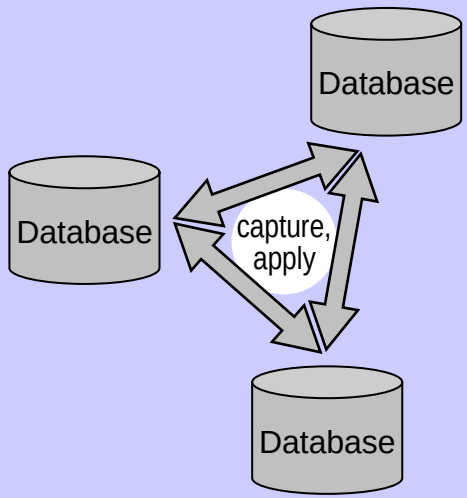
Information Integration – Data Patterns

EAI



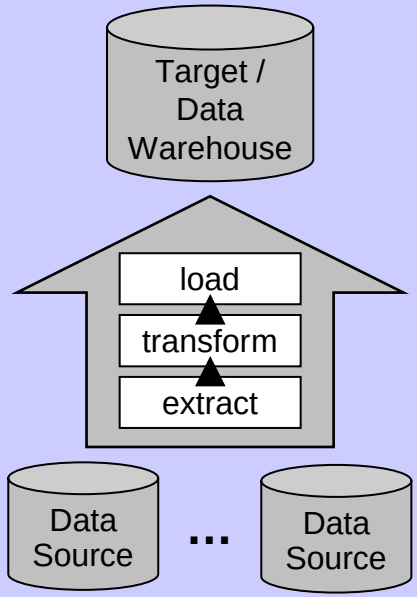
- Process based integration of application data
- Message-based, transaction-oriented processing
- Workflow and data orchestration, content-based routing

Replication



- Distribution, consolidation, or synchronization between databases
- Change capture provides event basis
- Apply controls batch or transactional and adds transformation

ETL



- Bulk data integration
- Set-based & hierarchical transformations
- High scale, batch-oriented data delivery

Replication, EAI or ETL?

Replication or EAI

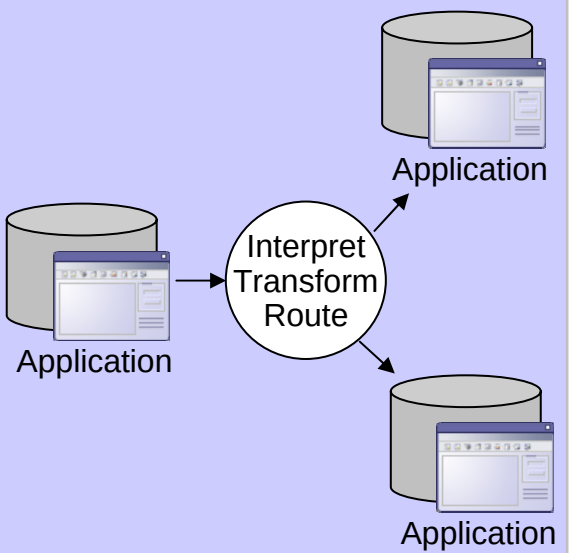
- Both can be used to accomplish the same task:
  - Capturing an event
  - Transferring it to another system
  - Applying it to the target application
- Reasons to use replication
  - Replication handles the end-to-end delivery process through declarative specification, including recovery processes.
  - The change capture (if log-based) occurs asynchronously from the originating application, reducing the performance impact on that application
  - The application is similarly shielded from any loss of availability of the message queue or transfer service
  - New or existing applications can be built without special coding for application messaging
- Reasons to use EAI:
  - The event being replicated is not written to a database
  - The event being replicated is not within the context of a single database
  - The target application requires invocation of the application interface to preserve integrity

Replication and/or ETL

- Both can be used to accomplish the same task:
  - Copying data from one or more databases to one or more databases
- Reasons to use replication
  - Want to capture only the changes (and the data does not contain sufficient information to for the extract process to identify changes)
  - Want lower latency between source and target systems
- Reasons to use ETL
  - Supports richer transformations
  - Easier to manage and maintain
  - Change volume is high, so full extract is easier and cheaper
- Reasons to use replication with ETL
  - Lower latency copies with richer transformations (typically limited to single row)

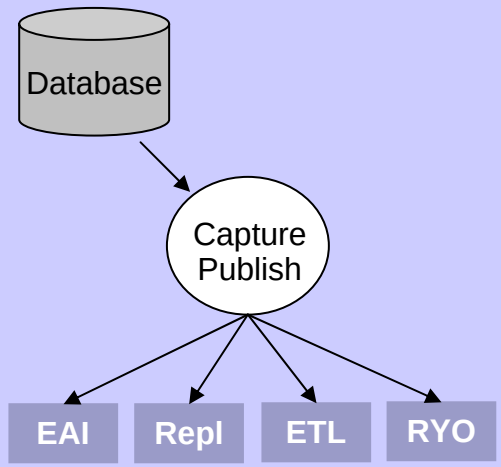
Information Integration – Data Patterns

EAI



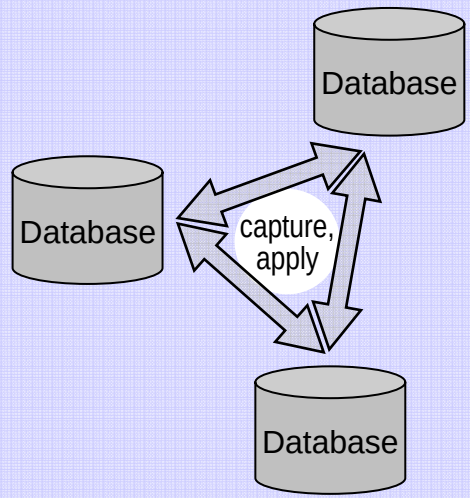
- Process based integration of application data
- Message-based, transaction-oriented processing
- Workflow and data orchestration, content-based routing

Data Event Publishing



- Message-based publishing based on event capture from single database
- Add-on to EAI, ETL, or Replication

Replication



- Distribution, consolidation, or synchronization between databases
- Change capture provides event basis,
- Apply controls batch or transactional and adds transformation

Event Publishing, EAI or ETL?

Event Publishing or EAI

- Both accomplish the same task:
  - Capturing an event
  - Putting a message about the event onto a queue
- Reasons to use event publishing
  - The messages are created asynchronously from the originating application, reducing the performance impact on that application
  - The application is similarly shielded from any loss of availability of the message queue or service
  - New or existing applications can be built without special coding for application messaging
- Reasons to use EAI:
  - Event is not written to a database
  - Database does not contain sufficient information for full message context
  - Message routing is based on message content
- Reasons to use event publishing with EAI
  - Simplest change capture for initiating a process or passing a message

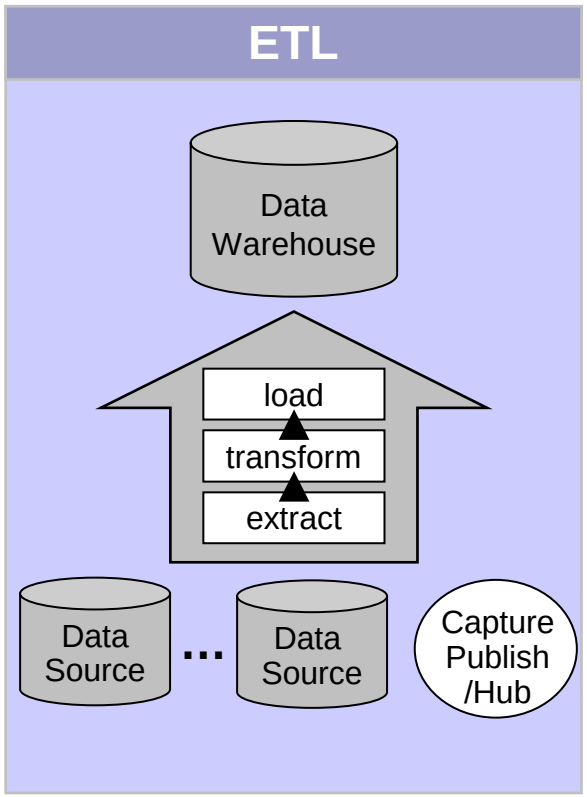
Event Publishing and ETL

- Reasons to use event publishing with ETL
  - Deliver changes to transformation engine and data flow

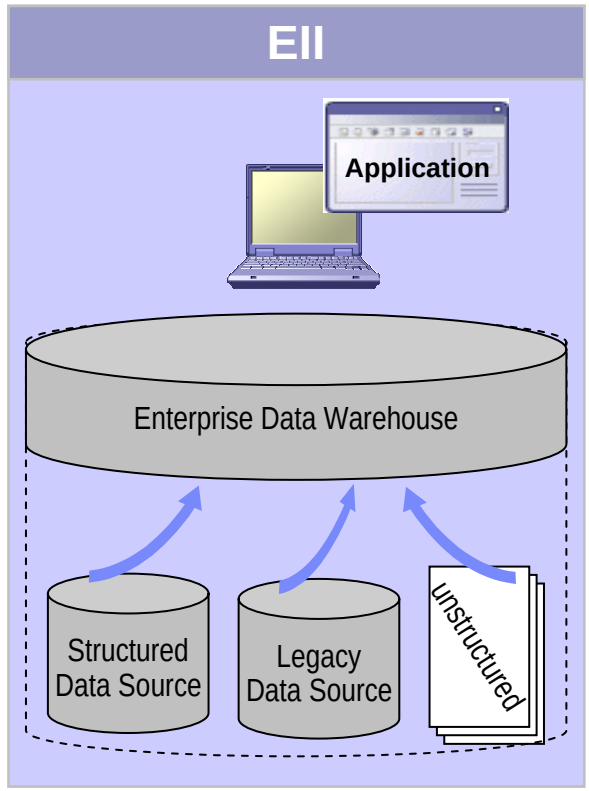
Event Publishing and Replication

- Reasons to use event publishing with Replication
  - Deliver changes to replication engine

Business Intelligence – Data Patterns



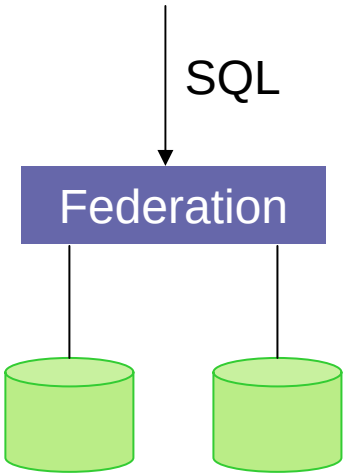
- Bulk data integration
- Set-based & hierarchical transformations
- High scale, batch-oriented data delivery



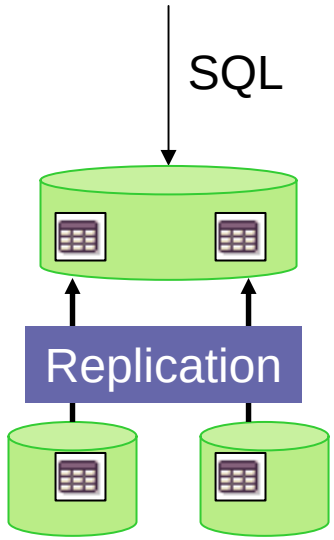
- Augmentation of Existing DW
- Real-time joins with data from multiple sources
- Dynamic drill down

Three types of data services

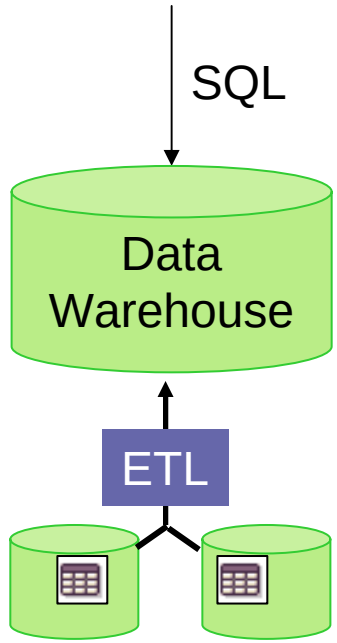
- Real-time



- Near Real-time



- Historical & analytical



Distributed Access vs. Consolidated Access

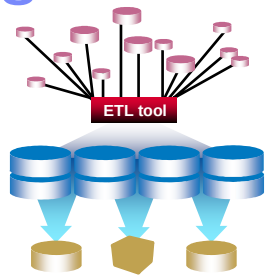
Distributed access	Consolidated access

Primary requirements:
▪ Very current data
▪ Dynamic joining of data
▪ Structured and unstructured data
▪ Mixed relational and non-relational data
▪ Not practical to copy data
▪ Small amounts of data in result set
	Primary requirements:
	▪ Local performance
	▪ Structured data
	▪ Extract, Transform and Load
	  ▪ Extensive transformations
	  ▪ User metadata
	  ▪ Large volumes of data
	▪ Replication
	  ▪ Small amount of changed data
	  ▪ Up to near real time updates

Agenda

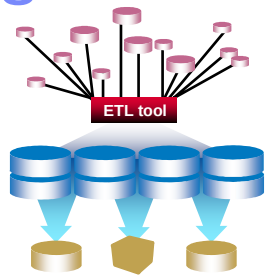
- Data Integration Challenges and IBM Vision
- Definitions and Patterns
- **Data Integration Approaches**
  - **ETL vs. EII vs. EAI**

ETL vs. EII vs. EAI – Strengths and Challenges



- ETL Major Strengths
  - Optimized for data structures
  - Periodic, batch-oriented (not intended for real-time)
  - Can move large volumes of data in one step
  - Enables complex data transformations requiring calculations, aggregations or multiple stages
  - Scheduling controlled by the administrator
  - Several GUI based tools available to increase productivity
  - High level of reuse of objects and transformations

ETL vs. EII vs. EAI – Strengths and Challenges



- ETL Main Challenges
  - Time to market
  - Change management
  - Data moved regardless of real need
  - Consumes storage systems
  - Data out-of-synch with the original source when it arrives in the DW
  - Large requirements for staging areas
  - Unidirectional
  - Lack of multi-site update support (2 phase commit)

ETL vs. EII vs. EAI – Strengths and Challenges

- EII Major Strengths
  - Relational access to non-relational sources
  - Ability to explore data before a formal data model and metadata are created
  - Quicker deployment
  - Can be reused by ETL and/or EAI further developments
  - Access in place data, meaning it avoids unnecessary movement of data.
  - Optimized for global access to remote sources
  - Event publishing technology provides a non-intrusive means to “listen” for particular changes (insert, update or deletes) that defined as being of interest.

ETL vs. EII vs. EAI – Strengths and Challenges

- EII Main Challenges
  - Need Matching keys across sources
  - Data types mismatch
  - Data reconciliation
  - Possibly high resource utilization on the source system
  - Limited to hundreds of thousands of rows for remote result sets
  - Performance degradation when query pushdown is not used
  - Limited transformation – bounded by SQL capability and system capacity
  - May consume network bandwidth during peak hours
  - Multi-site updates require transactional control (2PC)

ETL vs. EII vs. EAI – Strengths and Challenges

- EAI Major Strengths
  - Optimized for API-based applications
  - Real-time (or near)
  - Move/send individual events or transactions
  - Some capability for simple and basic transformation and rules
  - Workflow controlled
  - Broker capabilities (subscriptions)

ETL vs. EII vs. EAI – Strengths and Challenges

- EAI Main Challenges
  - Limited transformation capability
  - Limited support for data aggregation
  - Limited to tens of records per transaction
  - Complexity for development
  - Longer Time to market
  - Limited reuse for transformations
  - Limited support for metadata (use, import and export)
  - Semantic integrity
  - May consume network bandwidth during peak hours

ETL vs. EII vs. EAI – A Technical Comparison

	ETL	EII	EAI
Data Flow	Unidirectional – from source to target	Bidirectional	Bidirectional
Data Movement	Scheduled – batch Process managed	Query time Query (SQL) managed	Transaction triggered – asynchronous Transaction managed
Latency	Daily - Monthly	Real-time	Near real-time
Transformation, cleansing/enrichment Metadata process reuse	*Best* Generally high reusability of objects and processes	*Medium* Transformations embedded in views and database objects.	*Low* Transformations are done with ESQL. Metadata import limited with DB catalog information

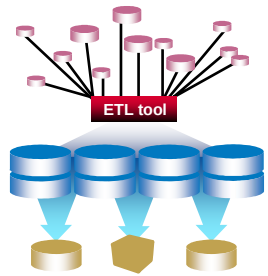
ETL vs. EII vs. EAI – A Technical Comparison

	ETL	EII	EAI
Transport	FTP, direct database connection	Direct database connection	Messaging
Data Volume Processing	Very large (millions, billions of records)	Medium – access to 100's of thousands or few millions of remote records	Small (few records) – can handle several parallel pipes of few records
Complexity of Transformation	Any complexity	Transformations that can be expressed with SQL	Simple syntax transformations. Limited semantic transformations can be implemented via a broker.

ETL vs. EII vs. EAI – A Technical Comparison

	ETL	EII	EAI
Support for Event Monitoring	Very limited with high latency	Limited to data events and depended on trigger capability of data sources	Best – logic can be added to support true event propagation and not only data transaction movement.
Versioning	Full support	Limited support – custom build	Limited support – custom build
Workflow Control	Scheduling, dependencies and error or exception handling	None	Extensive – rules based

ETL Best Practices



- ETL is usually heavily I/O bounded.
  - Avoid unnecessary staging steps
  - Use faster storage
  - Avoid I/O contention
  - Careful with lookup processing
  - Do not rely on ESS disk subsystems for file placement
- Use a tool and methodology for ETL, both for productivity and data consistency.
- Avoid populating data marts from data marts
- Avoid excessive locking
  - Key to running many concurrent processes in parallel
  - Allows backup at same time as query and load

EII Best Practices

- In general: **Don't** allow unregulated ad-hoc access
- Plan ahead to manage costly queries by caching frequently-used data at the DB2 II instance for best performance
- Control the kinds and cost of queries that are submitted to DB2 II using
  - Application controls: parameterized reports
  - DB2 Query Patroller
- Expect operations that move a lot of data between remote sources and DB2 II to take a long time
  - Don't try to use DB2 II to implement a "virtual warehouse" on a permanent basis, especially if ad-hoc access is desired

EII Best Practices

- Be mindful of the impact of federated queries on remote sources
  - Aim for “targeted” access to remote data
- Data flows from remote sources to the federated server
  - Be careful with joins of large tables residing in two or more remote systems
- Create and maintain a federated semantic layer

EAI Best Practices

- Avoid point-to-point integration
  - Use hub and spoke brokers for better reuse
- Design with deployment in mind
  - Build a deployment plan upfront
  - Establish SLAs and monitoring procedures
  - Understand the impact in all involved systems
  - Understand the data flow scenarios and contingencies
- Profile and monitor Performance
- Be prepare to track and trace data consistency and performance bottlenecks in the workflow

EII vs EAI vs ETL

- **When to use EII**
  - Usually connecting a large repository with selected data from other sources
  - *Selectively* as a tool to extend existing well-designed EDW
  - May be indicated when source data:
    - Volatility is high
    - Selectivity is granular
    - Connectivity is reliable
    - Service levels are compatible
    - Transformations are minimal and can be expressed as SQL
- **When to use EAI**
  - Integration of transactions and not large data sets
  - Questions can be answered by joining small amounts of data
  - Data sources repositories cannot be directly accessed
- **When to use ETL**
  - Data consolidation
  - Complex transformations

Combination is normally used