An Information Asset Hub

How to Effectively Share Your Data

Hello!

I am Jack Kennedy

Data Architect @ CNO
Enterprise Data Management Team
Jack.Kennedy@CNOinc.com

4 Data “Functions” Your Data Warehouse Does Today

Is the DW the right place?

Data Consolidation

- Bringing information entries from disparate source systems into a common location but in their original formats.
- RAW Data

Data Integration

- Translating and standardizing information entries to conform to common attribute and entity definitions
- Semantic Transformation

Raw Data

	Field Name	Data Type	Value
System A	Cust_Type	Char(1)	R
System A	Cust_Type	Char(1)	W
System B	Customer_Code	Varchar(5)	RTL
System B	Customer_Code	Varchar(5)	GOV

Integrated Data

Row #	Customer_Type	Source Name
1	Retail	Consolidated
2	Wholesale	System A
3	Government	System B

Data Historization

- Capturing snapshots of information entries each time a value changes

Historization			
Customer Name	**Customer Address**	**Effective From Date**	**Effective To Date**
John Wright	105 Riley Rd	09/05/2016	
John Wright	204 E Main St Apt A	01/05/2016	09/05/2016
John Wright	204 E Main St	05/10/2014	01/05/2016
John Wright	700 N State Dr	05/15/2013	05/10/2014

Change Data Capture

- Tracking data that has changed in the source so that change can be replicated in the target

Methods

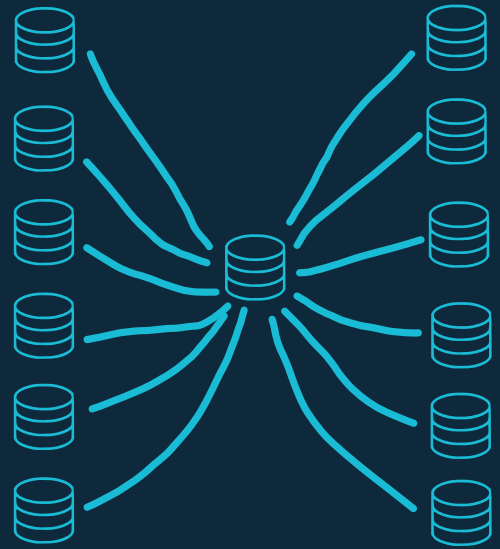
- 1. Full Difference Comparison
- 2. Source DB Triggers
- 3. Timestamp Tracking
- 4. Write Ahead Log Playback

Spaghetti Architecture

- S * T = 36

Hub & Spoke Architecture

- S + T = 12



Information Asset Hub

- Data sharing specialization
- Publish-Subscribe abstraction
- Purpose
  - Change Data Capture
  - Consolidation
  - Integration
  - Historization?
- Purposely NOT
  - NOT ad-hoc query
  - NOT analytics
  - NOT B/I
  - Let DW specialize for these

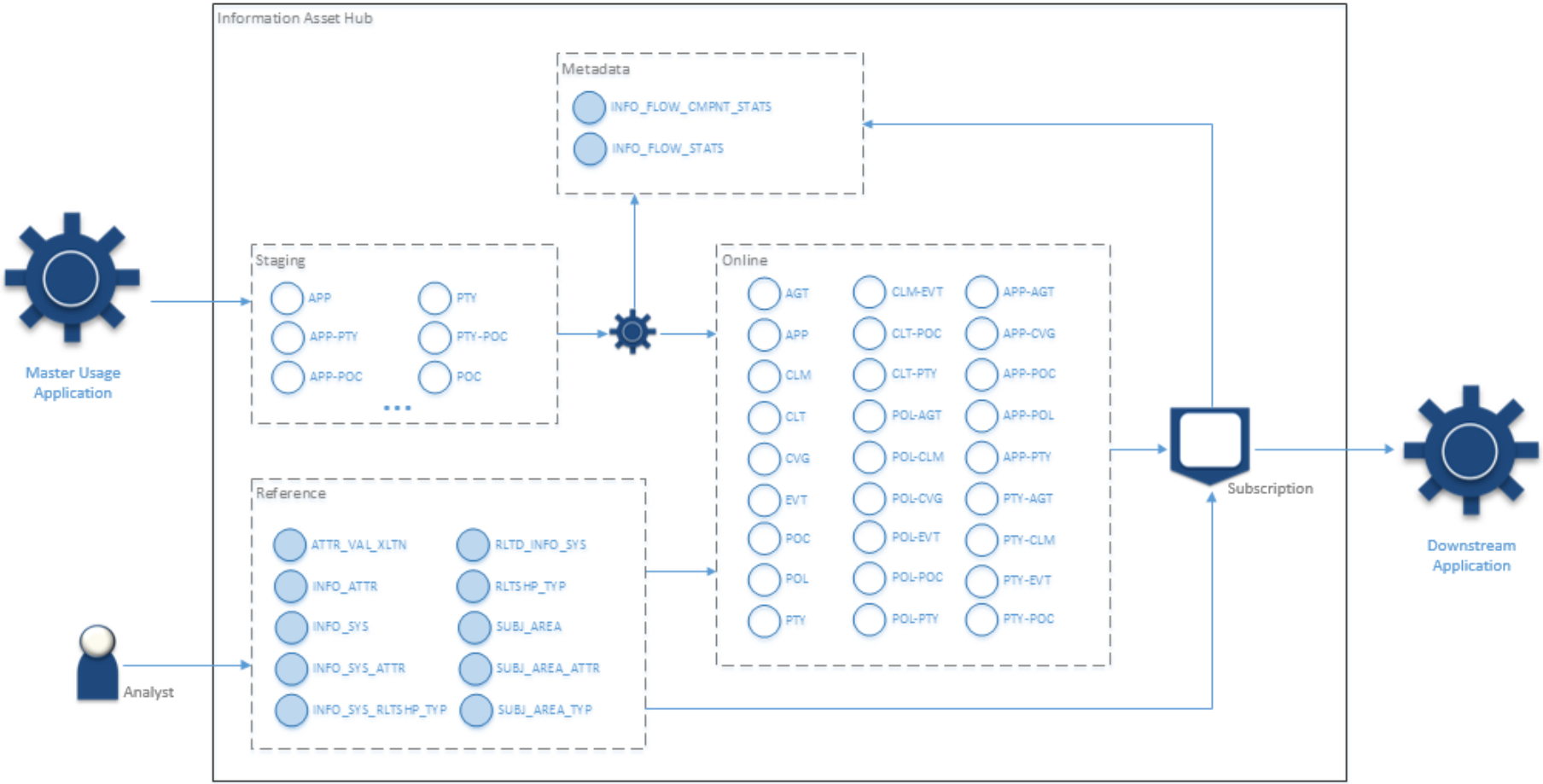
2

Our Implementation

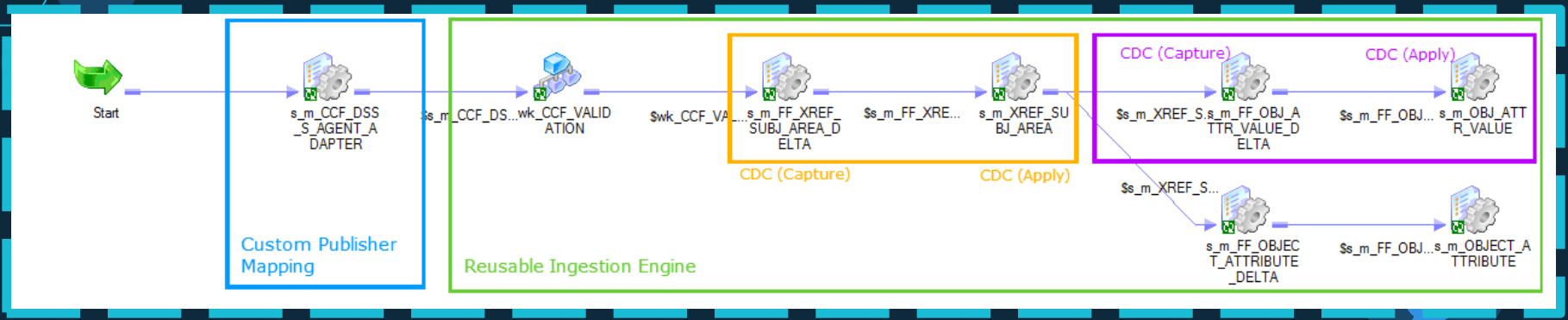
CURe (CNO's Unified Repository)

Approach + Technology

- Storage Platform: Oracle
  - KVP (Entity Attribute Value)
  - Row Based (New!)
- Change Data Capture: Oracle
  - Full Difference Comparisons
  - DB Pushdown – Full Outer Joins
- Consolidation: Informatica
- Integration: Informatica
- Subscription: Oracle Views as API


Information Asset Hub
Metadata
INFO_FLOW_CMPNT_STATS
INFO_FLOW_STATS
Master Usage Application
Staging
APP
APP-PTY
APP-POC
PTY
PTY-POC
POC
Online
AGT
APP
CLM
CLT
CVG
EVT
POC
POL
PTY
CLM-EVT
CLT-POC
CLT-PTY
POL-AGT
POL-CLM
POL-CVG
POL-EVT
POL-POC
POL-PTY
APP-AGT
APP-CVG
APP-POC
APP-POL
APP-PTY
PTY-AGT
PTY-CLM
PTY-EVT
PTY-POC
Subscription
Downstream Application
Reference
ATTR_VAL_XLTN
INFO_ATTR
INFO_SYS
INFO_SYS_ATTR
INFO_SYS_RLTSHP_TYP
RLTD_INFO_SYS
RLTSHP_TYP
SUBJ_AREA
SUBJ_AREA_ATTR
SUBJ_AREA_TYP
Analyst

Publish – KVP

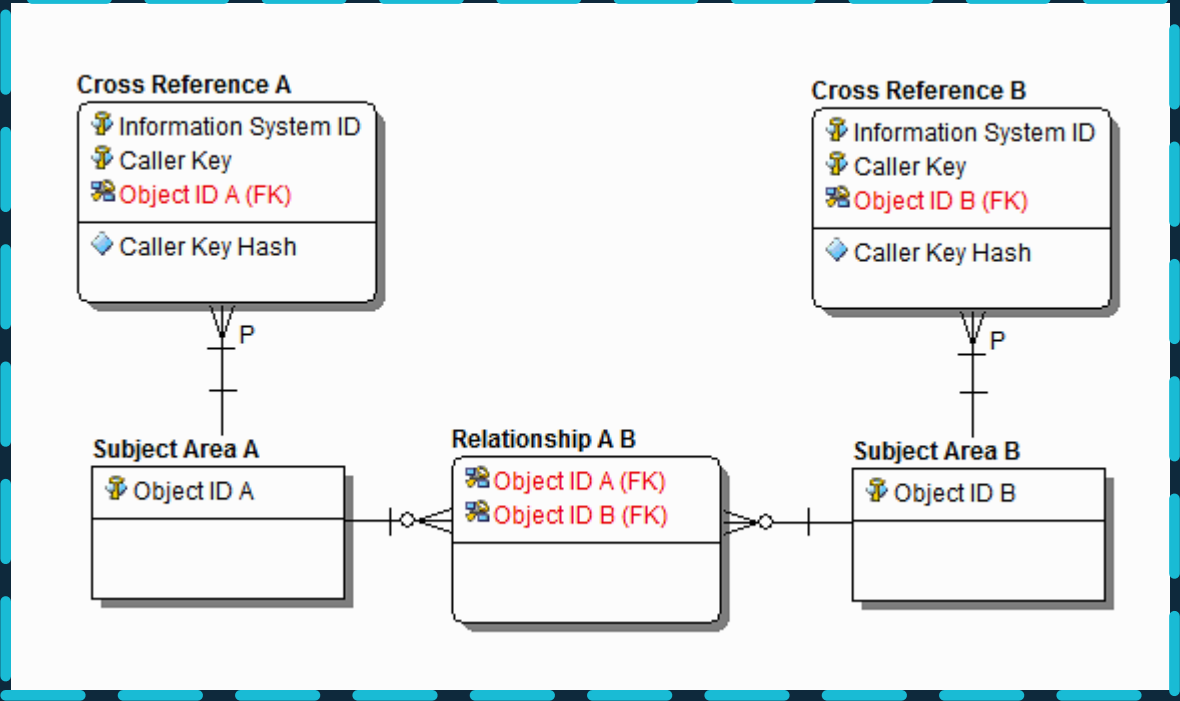


Src → CCF ⇄ Online
(unpivot) (CDC)

Publish – Row Stores

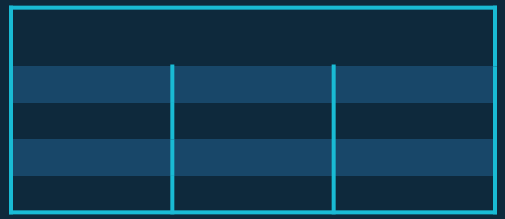
- NEW!
- Publishing Pattern Under Development
- Will Share Next Time (I Promise)

Data Model


Cross Reference A
Information System ID
Caller Key
Object ID A (FK)
Caller Key Hash
P
Cross Reference B
Information System ID
Caller Key
Object ID B (FK)
Caller Key Hash
P
Subject Area A
Object ID A
Relationship A B
Object ID A (FK)
Object ID B (FK)
Subject Area B
Object ID B

Subscribe

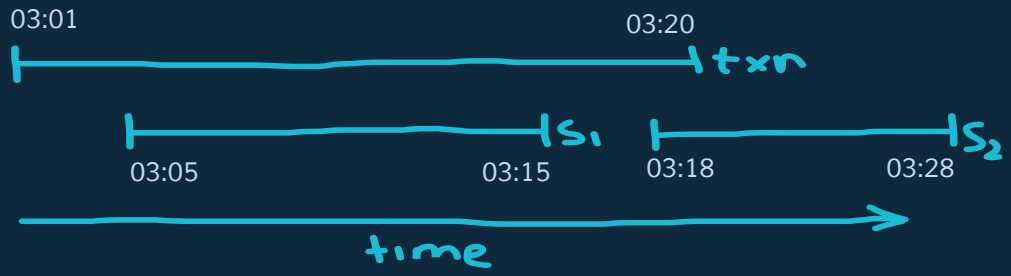
- Data driven configuration
- PL/SQL dynamically creates views per subscriber – access control
  - Full views give access to all records
  - Delta views gives access to changed records
- If KVP storage, pivot back to rows first
- UNION together KVP storage + row storage



```
create or replace view ...
as
select
```

Subscribe (Δ)

- Publisher tracks changes to records by updating timestamp audit fields
- Correct implementation example:



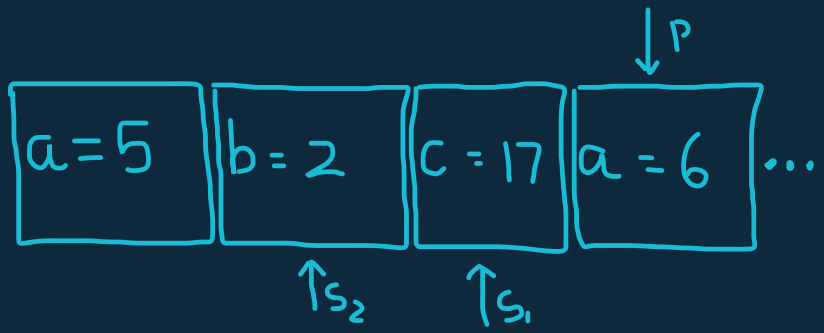
3

Big Data Implementation

The Log, Kafka, Avro, & Samza

The Log

- Append only file
- Ordered list of immutable “facts”
- Inside every database you’ve ever worked on
  - backup / recovery / replication
- Best physical structure for sharing data
  - Data is stored sorted in the order it changed over time – perfect for replication



Apache Kafka

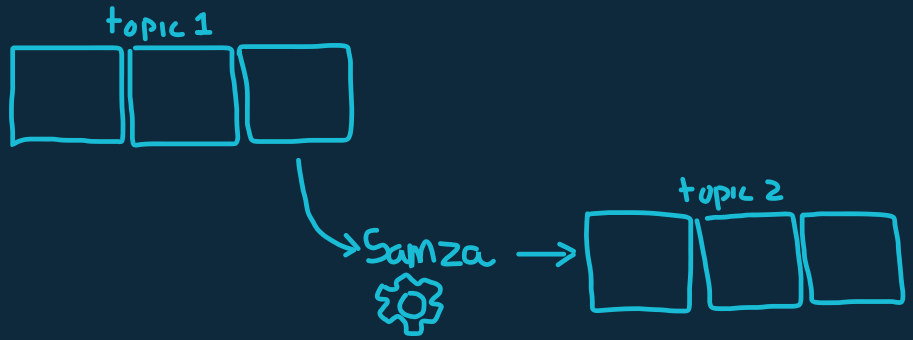
- http://kafka.apache.org/
- Horizontally scalable, fault tolerant, distributed log
- Each “stream” of data stored is called a Topic
- Producers publish data to topics
- Consumers subscribe from topics
- Compaction – removing “old” values
  - Time based – keep 2 weeks of history
  - Message based – for each message (row A), only keep the latest value
- This will work perfectly for our storage engine, but what about the storage format?

Apache Avro

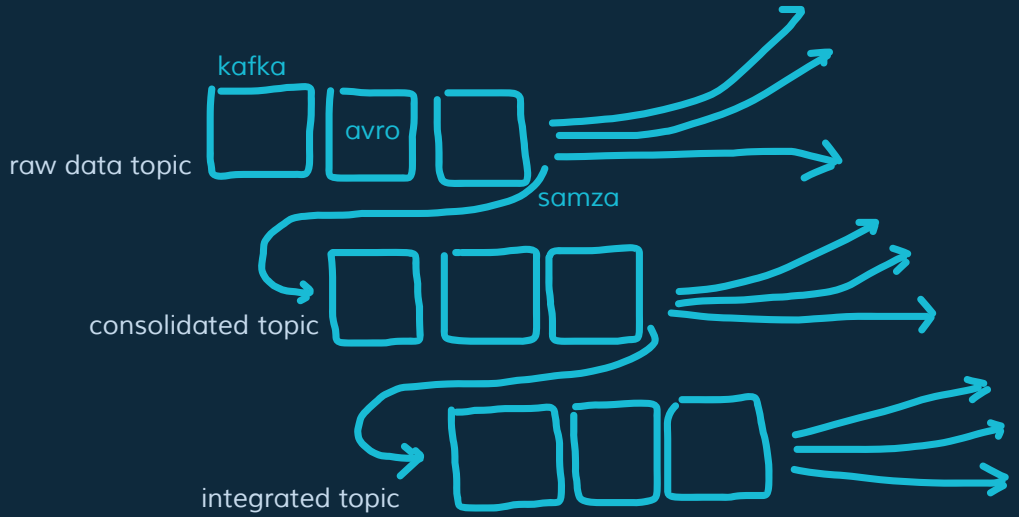
- https://avro.apache.org/
- Data serialization system
  - Saving an object to disk as a file
- Compact, fast, binary data format
- JSON used to define the “schema”
- Schema can be persisted with the data, or stored in a library
- Supports schema evolution
  - Readers only need to know the writer’s schema
  - Uses conflict resolution rules to translate from writer’s schema to what the reader initially expected
- Now we have the storage engine, and the storage format, but how do we process / transform the data

Apache Samza

- http://samza.apache.org/
- Distributed stream processing framework
- Built from the ground up to work with Kafka topics
- Samza Job – code that performs a logical transformation on a set of input streams to append output messages to a set of output streams



Big Data Information Asset Hub



Thanks!

Any questions?

You can find me at:

- Jack.kennedy@CNOinc.com

Credits

Special thanks to all the people who made and released these awesome resources for free:

- Presentation template by SlidesCarnival
- Photographs by Unsplash

Resources

To learn more, please check out the following GREAT resources:

◇https://www.ibm.com/developerworks/community/blogs/SusanVisser/entry/new_book_patterns_of_information_management?lang=en

◇http://dataintensive.net/