Real-Time Data Warehousing L15 (Lec 22)

Topics

- What is Real-Time Data Warehousing (RTDWH)?
- Operational Data Source (ODS)
- Why we need RTDWH?
- Challenges
- Solutions

What is RTDWH?

- Business Intelligence (BI) applications and their underlying data warehouses have been used primarily as *strategic* decision-making tools
- Kept Separate from Operational systems that manage day-to-day business operations
- Significant industry momentum toward using BI for driving *tactical* day-to-day business decisions and operations

What is RTDWH?

- The concept and application of data warehousing solutions has progressed rapidly since its inception in the mid-eighties.
- DW initiative was driven by the different silos of information that exists with the different departments or functional groups within a company
- Legacy applications were built on different technology platforms with different communication and data transmission protocols.
- This made it difficult and time consuming to get a single, cohesive view of the information users needed to be productive.
- Getting access to this information required a process that was typically only done on a periodic basis.

What is RTDWH?

- Data warehousing has progressed rapidly to the point that real-time data warehousing is now the focus of CIO’s across the globe
- Real-time data warehousing requires a solid approach to data integration and most importantly, the ability to transform and filter data on-the-fly to ensure it meets the needs of its different users

Why Real Time Data Warehousing?

- Active decision support
- Business activity monitoring (BAM)
- Alerting
- Efficiently execute business strategy
- Monitoring is completed in the background
- Positions information for use by downstream applications
- Can be built on top of existing data warehouse

Why Real Time Data Warehousing?

- Up-to-the-moment reporting against a twinkling DB
- Business users need to access production applications that run the business
- Users need to access two different systems:
  - DW for historical picture of what happened in the past
  - Many OLTP systems for what is happening today

Why Real Time Data Warehousing?

- Why can’t we get all the business information in one place?
- Typically in a DW the latency is 24 hours
- For fast moving vertical industries, this delay is too much

Why Real Time Data Warehousing?

- DW too has become mission critical
- Feeds *enriched* information back to operational systems that is then used to:
  - Process transactions
  - Personalize offers
  - Present up-sell promotions
- Push for ever-fresher information is ON

Why Real Time Data Warehousing?

- Some other factors that have forced DW to change:
  - CRM
  - Zero-latency enterprise business deal
  - Globalization & the Web

Why Real Time Data Warehousing?

- CRM
  - Modern CRM demands a contemporary, consistent, & complete profile of the customer available to all operational systems that directly or indirectly serve the customer
  - DWs need constant customer information streams from operations
  - But, increasingly operational systems rely on DW enrichment of customer information
  - Architectural alternatives need to be explored that can support more generalized integration scenario – between OLTP & DW with ever increasing urgency

Why Real Time Data Warehousing?

- Zero-latency enterprise business deal
  - Exhorts the benefit of speed & a single version of the truth
  - In a real-time, zero-latency enterprise, information is delivered to the right place at the right time for maximum business value
  - Right-time Systems
  - DWs are under pressure to provide low-latency view of the health of the business

Why Real Time Data Warehousing?

- Globalization & the Web
  - 24x7 businesses and round the clock access to DW
  - Coupled with the need to warehouse more & more data
  - Time window available to load the DW compressed
  - Challenge to the ETL team
  - Can’t we somehow trickle-feed the DW throughout the day, rather than trying to shoehorn expanding data loads into shrinking windows of acceptable downtime

Real-Time ETL

- Tool that moves data asynchronously into a DW with some urgency – within minutes of execution of the business Tx
- RTDWH demands a different approach to ETL methods used in batch-oriented DW
- Running ETL batches more frequently is not practical either to OLTP or to DW
- Including the DW in the commit logic doesn’t work either
- Locking & 2-phase commit also doesn’t work across systems with different structures & granularity

Real-Time ETL

- ETL system has a well defined boundary where dimensionally prepared data is handed over to the front room
- A real-time system cannot have this boundary
- Architecture of front-end tools is also affected at the same time
- 3 data delivery paradigms that require an end-to-end perspective (from original source to user's screen)
  - Alerts
  - Continuous polling
  - Non-event notification

Real-Time ETL

- Alerts
  - A data condition at the source forces an update to occur at the user's screen in real time
- Continuous polling
  - The end user's application continuously probes the source data in order to update the user's screen in real-time
- Non-event notification
  - The end user is notified if a specific event does not occur within a time interval or as the result of a specific condition

Traditional Vs. Real-Time Data Warehouse

- Traditional Data Warehouse (EDW)
  - Strategic
    - Passive
    - Historical trends
  - Batch
    - Offline analysis
  - Isolated
    - Not interactive
  - Best effort
    - Guarantees neither availability nor performance

Traditional Vs. Real-Time Data Warehouse

- Real-Time Data Warehouse (RTDWH)
  - Tactical
    - Focuses on execution of strategy
  - Real-Time
    - Information on Demand
    - Most up-to-date view of the business
  - Integrated
    - Integrates data warehousing with business processes
  - Guaranteed
    - Guarantees both availability and performance

Real-Time Integration

- Goal of real-time data extraction, transformation and loading
  - Keep warehouse refreshed
  - Minimal delay
- Issues
  - How does the system identify what data has been added or changed since the last extract
  - Performance impact of extracts on the source system

RTDWH Lineage

- Operational Data Source (ODS)
- Motivations of the original ODS were similar to modern RTDWH
- Implementation of RTDWH reflects a new generation of SW/HW & techniques

RTDWH

- RTDWH advocates that instead of pulling operational data from OLTP system in nightly batch jobs into an ODS, data should be collected from OLTP systems as and when events occur move them directly into the data warehouse.
- This enables the data warehouse to be updated instantaneously and removes the necessity of an ODS.

RTDWH

- Tactical and strategic queries can be fired against this RTDWH to use immediate as well as historical data.
- Some proponents go even further to propose that data marts are redundant and analytic queries can be fired against the data warehouse with slight adjustments.
- Instead of having the earlier topology of an ODS, a data warehouse and data marts in separate systems, put everything in one big box called the DW which houses real-time data for tactical queries, historic data for strategic queries and segregated data for analysis groups.

RTDWH

- Current EAI tools provide the opportunity to pull real-time data out of OLTP systems and pump it into large data warehouses

A Word About ODS

- ODS is also referred to as Generation 1 DW
- Separate system that sat between source transactional system & DW
- Hot extract used for answering narrow range of urgent operational questions like:
  - Was the order shipped?
  - Was the payment made?
- ODS is particularly useful when:
  - ETL process of the main DW delayed the availability of data
  - Only aggregated data is available

A Word About ODS

- ODS plays a dual role:
  - Serve as a source of data for DW
  - Querying
- Supports lower-latency reporting through creation of a distinct architectural construct & application separate from DW
- Half operational & half DSS
- A place where data was integrated & fed to a downstream DW
- Extension of the DW ETL layer

A Word About ODS

- ODS has been absorbed by the DW
  - Modern DWs now routinely extract data on a daily basis
  - Real-time techniques allow the DW to always be completely current
  - DWs hav become far more operational than in the past
  - Footprints of conventional DW & ODS now overlap so completely that it is not fruitful to make a distinction between the kinds of systems

A Word About ODS

- **Classification of ODS based on:**
  - **Urgency**
    - **Class I - IV**
  - **Position in overall architecture**
    - **Internal or External**

A Word About ODS

Classifications	Comment
Class 1	▪ Refresh cycle: Real-time ▪ Degree of transformation: Low due to compressed timeframes
Class 2	▪ Refresh cycle: 1/2 - 1 hour store and forward mechanism ▪ Degree of transformation: Medium
Class 3	▪ Refresh cycle: Daily. Traditional batch process ▪ Degree of transformation: High
Class 4	▪ Refresh cycle: Ad hoc, often involving preprocessed, value-added information from a data warehouse ▪ Degree of transformation: Highest

A Word About ODS

- Urgency
  - Class I – Updates of data from operational systems to ODS are synchronous
  - Class II – Updates between operational environment & ODS occurs between 2-3 hour frame
  - Class III – synchronization of updates occurs overnight

A Word About ODS

- Urgency
  - Class IV – Updates into the ODS from the DW are unscheduled
    - Data in the DW is analyzed, and periodically placed in the ODS
    - For Example –Customer Profile Data
    - Customer Name & ID
    - Customer Volume – High/low
    - Customer Profitability – High/low
    - Customer Freq. of activity – very freq./very infreq.
    - Customer likes & dislikes

Concluding Remarks

- If an ODS exists (Type 1), then it can contribute towards RTDWH
- If it does not exist, then there is no point in building an ODS for either conventional DW or for RTDWH

Kimball's Approach to RTDWH

- Real-Time Partitions (Generation 2)
  - Separate real-time fact table is created whose grain & dimensionality matches that of the corresponding FT in the static (nightly loaded) DW
  - Real-time FT contains only current day's facts (those not yet loaded into the static FT)
  - Each night, the contents of RTFT are written to the static FT and the RTFT is purged, ready to receive the next day's facts

Kimball's Approach to RTDWH

- Real-Time Partitions
  - Gives RT reporting benefits of the ODS into the DW itself, eliminating ODS architectural overhead
  - Facts are trickled into the RTFTs throughout the day
  - User queries against the RTFTs are neither halted nor interrupted by this loading process

Kimball's Approach to RTDWH

- Real-Time Partitions
  - Indexing is minimal
  - Performance is achieved by restricting the amount of data in RTFTs
  - Caching entire RTFT in memory
  - Create view to combine data from both static & real-time FT, providing a virtual star schema to simplify queries that demand views of historical measures that extend to the moment

Kimball's Approach to RTDWH

- Real-Time Partitions
  - Fact records alone trickled into RTFT
  - Any issues?
  - What about the changes to DTs that occur between the nightly bulk loads?
  - New customers created during the day!
  - Are we focusing only on fresh facts?

Kimball's Approach to RTDWH

- Real-Time Partitions
  - Hybrid approach to SCD in real time environment
    - Treat intra-day changes to a DT as TYPE 1, where a special copy of the DT is associated with the RT partition
    - Changes during the trigger simple overwrites
    - At the end of the day, any such changes can be treated as TYPE 2 in the original DT

Real-Time ETL

- Microbatch ETL
- Enterprise Application Integration (EAI)
- Capture, Transform, & Flow (CTF)
- Enterprise Information Integration (EII)
- Real-Time Dimension Manager

Challenge #1: Enabling Real-time ETL

- Solution 1a: "Near Real-time" ETL
- Solution 1b: Direct trickle feed
- Solution 1c: Trickle & Flip
- Solution 1d: External Real-time Data Cache

Challenge #2: Modeling Real-time Fact Tables

- Solution 2a: Modeling as Usual with Direct Fact Table Feed
- Solution 2b: Separate Real-time Partition
- Solution 2c: Integrated Real-time through Views
- Solution 2d: Modeling with an External Real-time Data Cache

Challenge #3: OLAP Queries vs. Changing Data

- Solution 3a: Use a Near Real-time Approach
- Solution 3b: Risk Mitigation for True Real-time
- Solution 3c: Use an External Real-time Data Cache

Challenge #4: Scalability & Query Contention

- Solution 4a: Simplify and Limit Real-time Reporting
- Solution 4b: Apply More Database Horsepower
- Solution 4c: Separate & Isolate in a Real-time Data Cache
- Solution 4d: Just-in-time Information Merge from External Data Cache
- solution 4e: Reverse Just-in-time Data Merge

Challenge #5: Real-time Alerting

- Solution 5a: n-Minute Cycle Schedule
- Solution 5b: True real-time data monitoring & triggering
- Solution 5c: Real-time Alert Threshold Management