CDC (Change Data Capture) – Comprehensive Explanation

Unit I: Introduction to CDC

Definition of Change Data Capture

Change Data Capture is a design pattern that identifies, captures, and delivers individual data changes occurring in source systems in real-time or near-real-time. Instead of processing entire datasets, CDC monitors only modified data—inserts, updates, and deletes—and propagates these changes to downstream systems.

Real-time Example: When a customer updates their shipping address on an e-commerce website, CDC immediately captures this single change and sends it to the warehouse inventory system, customer analytics dashboard, and shipping partner system, ensuring all systems reflect the new address without processing the entire customer database.

Need for CDC in Modern Systems

Modern systems require fresh data across distributed architectures. CDC addresses the need for:

- Real-time decision making
- Microservices data synchronization
- Zero-downtime migrations
- Regulatory compliance (audit trails)
- Reduced latency in data pipelines

Real-time Example: A financial trading platform uses CDC to synchronize user portfolio data across risk calculation engines, compliance systems, and trading dashboards in milliseconds, enabling real-time trading decisions while maintaining data consistency.

Problems with Traditional Data Integration

Traditional ETL batch processing suffers from:

- High latency (hours/days old data)
- Performance overhead on source systems
- Complex incremental load logic
- Data inconsistency windows
- Inability to capture deletes

Real-time Example: A retail chain using nightly batch updates discovers inventory discrepancies where online sales aren't reflected in store inventory systems, leading to overselling. CDC solves this by streaming inventory changes immediately.

CDC vs Batch Processing

CDC provides continuous, low-latency data movement vs. scheduled batch windows:

- **CDC:** Event-driven, near-real-time, low-impact
- **Batch:** Schedule-driven, high-latency, high-impact

Real-time Example: A healthcare provider switching from daily batch patient data updates to CDC reduced medication error risks by updating patient records across all systems within seconds of lab result updates.

Real-world Use Cases of CDC

1. **Real-time Analytics:** Clickstream analysis for personalized recommendations
2. **Data Lake/Warehouse Updates:** Continuous freshness in Snowflake/BigQuery
3. **Database Migration:** Zero-downtime cloud migrations
4. **Microservices Synchronization:** Maintaining materialized views
5. **Audit Compliance:** Complete change history tracking

Real-time Example: Uber uses CDC to stream driver location updates, ride status changes, and payment transactions to their real-time analytics platform for dynamic pricing and dispatch optimization.

Unit II: Types of CDC Techniques

Trigger-Based CDC

Uses database triggers to capture changes by writing to shadow tables when DML operations occur.

Real-time Example: A legacy banking system uses triggers on the `transactions` table to populate a `transaction_audit` table, which is then consumed by their fraud detection system. Each INSERT/UPDATE fires a trigger that logs the change with timestamp and user context.

Timestamp-Based CDC

Relies on `LAST_UPDATED` or `CREATED_AT` columns to identify changed records since the last extraction.

Real-time Example: A sales CRM system extracts all contacts modified in the last 15 minutes using `modified_timestamp > last_extraction_time` to update marketing automation platforms, but misses hard deletes without timestamps.

Snapshot-Based CDC

Compares current and previous table snapshots using checksums or full comparisons to identify differences.

Real-time Example: A healthcare analytics platform compares daily snapshots of patient visit tables to detect new admissions, but requires significant storage and processing for large datasets.

Log-Based CDC

Reads database transaction logs (redo logs, WAL, binlog) to capture changes at the lowest level without impacting application performance.

Real-time Example: Netflix uses MySQL binlog CDC to stream view history changes to their recommendation engine, capturing every pause, play, and rating in real-time without affecting the user experience database.

Comparison of CDC Techniques

- **Performance:** Log-based (best), Trigger-based (worst)
- **Completeness:** Log-based (captures all changes including deletes)
- **Implementation Complexity:** Snapshot-based (simplest), Log-based (requires deep DB knowledge)
- **Latency:** Log-based (lowest), Snapshot-based (highest)

Real-time Example: Amazon uses log-based CDC for their order processing system because triggers would degrade performance during Prime Day sales spikes, while timestamp-based would miss critical intermediate states.

Unit III: CDC Architecture and Workflow

CDC System Architecture

A typical CDC architecture includes:

1. Source Database
2. Change Capture Mechanism
3. Message Broker/Event Stream
4. Change Processors
5. Target Systems

Real-time Example: Airbnb's CDC architecture captures listing availability changes from MySQL, streams via Kafka, processes with Flink, and updates Elasticsearch for search, Cassandra for recommendations, and Redshift for analytics.

Source Database

The origin system where operational data changes occur (RDBMS, NoSQL, mainframe).

Real-time Example: Walmart's inventory management system using PostgreSQL where stock level changes originate during checkout transactions.

Change Capture Mechanism

The component that detects and extracts changes using one of the CDC techniques.

Real-time Example: Debezium connectors reading Oracle redo logs to capture order status changes in real-time for FedEx's tracking system.

Event Streaming Layer

Message brokers (Kafka, Pulsar) that buffer and distribute change events.

Real-time Example: LinkedIn uses Apache Kafka to stream profile update changes from their database to 150+ downstream services with varying consumption rates and requirements.

Target Systems

Destination systems consuming change events:

- **Data Warehouses:** Snowflake, Redshift
- **Data Lakes:** S3, ADLS
- **Search Indices:** Elasticsearch
- **Microservices:** GraphQL layers, gRPC services

Real-time Example: When a GitHub repository setting changes, CDC updates: 1) Elasticsearch for search, 2) Redis cache for API responses, 3) Data warehouse for analytics, and 4) Notification service for subscribers.

End-to-End CDC Data Flow

1. Change occurs in source DB
2. Capture mechanism reads transaction log
3. Event serialized (Avro/Protobuf)
4. Published to message broker
5. Consumers process and apply changes
6. Monitoring/alerting tracks latency

Real-time Example: When a Tesla vehicle reports new telemetry: 1) Aurora PostgreSQL updates, 2) AWS DMS captures change, 3) Kinesis streams event, 4) Lambda processes for

predictive maintenance, 5) DynamoDB updates vehicle state, 6) CloudWatch monitors pipeline health.

Unit IV: Tools and Technologies for CDC

Database-Native CDC Tools

Built-in CDC capabilities within database systems.

Oracle GoldenGate

Comprehensive real-time data integration platform.

Real-time Example: A global bank uses GoldenGate to replicate transactions between on-prem Oracle databases and AWS RDS for Oracle during their cloud migration, maintaining sub-second latency across continents.

SQL Server CDC

Native change tracking using SQL Server Agent and change tables.

Real-time Example: A hospital uses SQL Server CDC to track patient record changes for HIPAA compliance audits, automatically capturing before/after values of sensitive data modifications.

Open-Source CDC Tools

Debezium

Distributed CDC platform built on Kafka Connect.

Real-time Example: Shopify uses Debezium to stream MySQL inventory changes to Kafka, enabling real-time inventory synchronization across their global fulfillment network.

Apache Kafka

Not a CDC tool itself but provides Connect API for source connectors.

Real-time Example: PayPal uses Kafka Connect with MySQL and MongoDB connectors to stream payment events to their fraud detection and analytics systems at scale (millions of events per second).

Cloud-Based CDC Solutions

AWS DMS

Managed database migration and replication service.

Real-time Example: Zoom uses AWS DMS for continuous replication from their production PostgreSQL databases to Redshift for real-time analytics on meeting participation patterns.

Azure Data Factory

Cloud-based data integration with CDC capabilities.

Real-time Example: BMW uses Azure Data Factory's CDC capabilities to synchronize vehicle sensor data from factory databases to Azure Synapse for predictive quality analysis.

CDC in Microservices Architecture

CDC enables data sovereignty while sharing changes via events.

Real-time Example: Uber Eats uses CDC to propagate restaurant menu changes from the "Restaurant Service" database to the "Search Service" Elasticsearch index and "Pricing Service" cache, maintaining eventual consistency without direct DB access.

Unit V: Advantages, Challenges, and Applications

Advantages of CDC

1. **Low Latency:** Near-real-time data availability
2. **Source Efficiency:** Minimal performance impact
3. **Reliability:** Transactionally consistent changes

4. **Completeness:** Captures all change types including deletes
5. **Historical Tracking:** Enables point-in-time recovery

Real-time Example: Stripe's financial reporting system uses CDC to achieve sub-minute reconciliation between transaction processing systems and reporting databases, enabling real-time financial monitoring.

Challenges and Limitations

1. **Schema Changes:** Breaking changes require pipeline updates
2. **Initial Load:** Historical data synchronization complexity
3. **Data Volume:** High-frequency changes can overwhelm systems
4. **Debezium vs Tool-Specific:** Vendor lock-in considerations
5. **Monitoring Complexity:** End-to-end latency tracking

Real-time Example: Twitter struggled with schema evolution when adding new tweet metadata fields, requiring careful coordination between application deployments and CDC pipeline updates.

CDC and Data Consistency

CDC maintains transactional integrity through:

- Event ordering guarantees
- Exactly-once or at-least-once semantics
- Dead letter queue handling
- Idempotent consumers

Real-time Example: Financial exchanges use CDC with idempotent consumers and transaction ID tracking to ensure trade settlement systems process each trade exactly once, even during network retries.

CDC in Real-Time Analytics

Enables fresh analytics without batch windows.

Real-time Example: Instagram uses CDC to stream engagement metrics (likes, comments, shares) to their real-time analytics dashboard, allowing content moderators to detect trending violations within seconds.

CDC in Data Replication and Synchronization

Maintains consistent copies across distributed systems.

Real-time Example: Salesforce uses CDC to replicate customer data changes from their core multi-tenant database to regional read replicas, ensuring low-latency access for global users while maintaining consistency.

Future Trends in CDC

1. **ML-Enhanced CDC:** Anomaly detection in data streams
2. **Serverless CDC:** Fully managed, auto-scaling pipelines
3. **Multi-Model CDC:** Unified capture across SQL/NoSQL/streams
4. **Edge CDC:** IoT and edge computing integration
5. **Blockchain Integration:** Immutable change audit trails

Real-time Example: Future autonomous vehicle networks may use edge CDC to stream partial data summaries from vehicles to central ML models while maintaining full audit trails of sensor data changes for regulatory compliance.

Conclusion

CDC has evolved from a niche replication technique to a foundational pattern for modern data architectures. By providing real-time, efficient, and reliable data movement, CDC enables organizations to build responsive systems that can react to changes as they happen. The choice of CDC technique depends on specific requirements around latency, performance impact, and infrastructure constraints, with log-based CDC increasingly becoming the standard for demanding production systems.