On Building Integrated and Distributed Database Systems

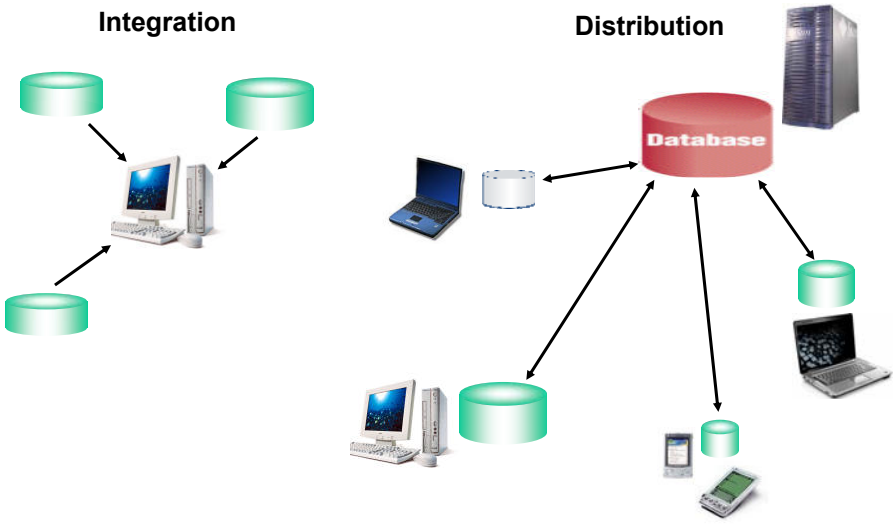
Data Replication

Robert Wrembel
Poznań University of Technology
Institute of Computing Science
Poznań, Poland
`Robert.Wrembel@cs.put.poznan.pl`
`www.cs.put.poznan.pl/rwrembel`

Data distribution

- Aims of data distribution
- Data replication: problems and solutions
- Data replication in commercial DBMSs
  - Oracle
  - Microsoft SQL Server
  - IBM DB2

Integration and distribution

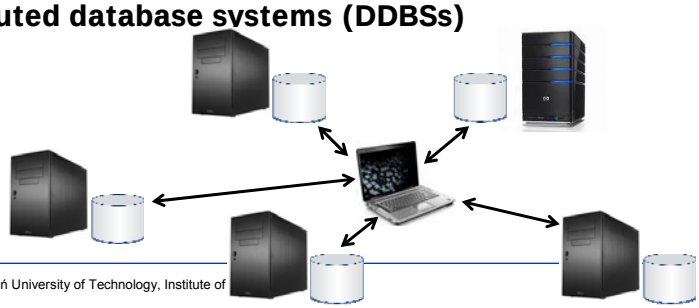


Data distribution aims

- Mobile devices
  - increasing data availability
  - distributed collecting of data ⇨ need of synchronizing with a central DB
  - distributed processing / analysis of data
  - traveling salesmen / women

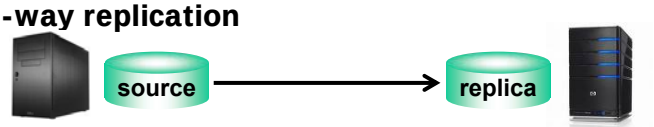
Data distribution aims

- Increasing processing efficiency
  - load balancing
  - processing parallelism
  - reducing data access time over network ⇨ data close to their "consumers"
- Increasing data availability in the case of system or network failure
- Distributed database systems (DDBSs)

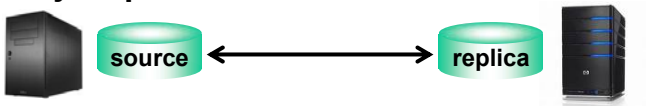


Data replication

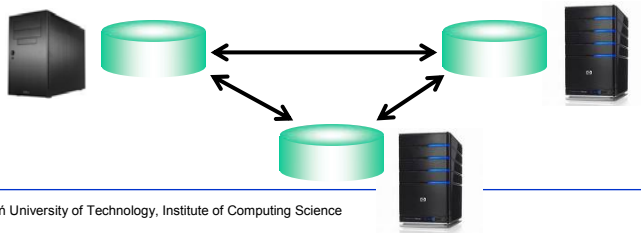
- One-way replication



- Two-ways replication



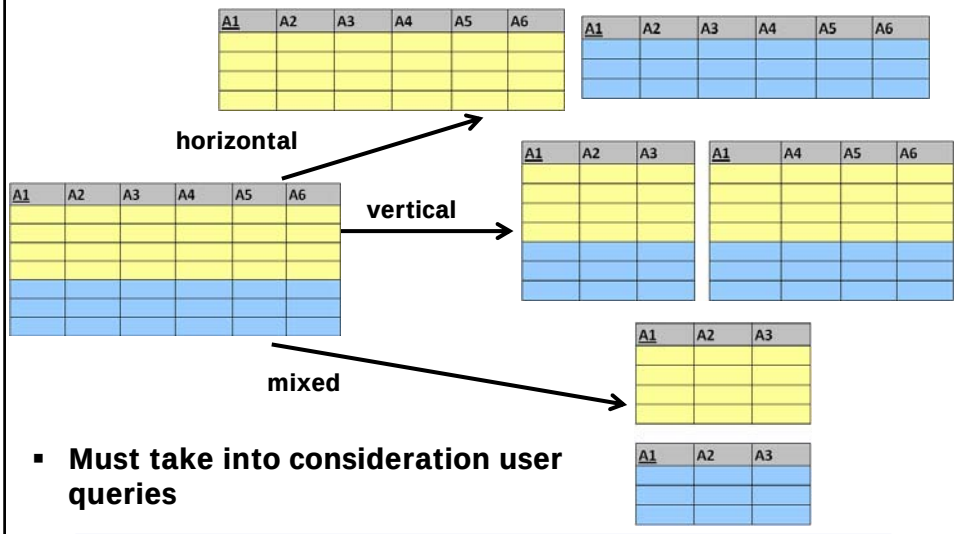
  - Multi-master replication



Data replication problems

- What to replicate?
- Refreshing replicas
  - when to refresh?
  - how to refresh?
  - what to send?
  - how to detect changes?
- Into which nodes to replicate?
- How to balance the load between nodes?

What to replicate?



- Must take into consideration user queries

Refreshing replicas

- Refreshing replicas
  - when to refresh?
    - synchronously
    - asynchronously - automatically
    - asynchronously - on demand
  - how to refresh?
    - fully
    - incrementally
  - what to send?
    - data shipping (Oracle, SQL Server)
    - transaction shipping (SQL Server, DB2)
  - how to detect changes?

How to detect changes?

- Audit columns
  - additional columns: operation timestamp, operation type
  - updated by: db triggers or applications
- Logging operations
  - user-implemented log
  - system log (e.g., Oracle snapshot log)
- Comparing two consecutive snapshots of a data source
- DB triggers for synchronous replication
- Analysis of a redo log (transaction log)
  - periodically (log scraping) / on-line (log sniffing)

Into which nodes to replicate?

- Allocation algorithm has to take into consideration
  - characteristics of queries in nodes
  - data transmission (replica refreshing) costs between data
  - storage costs
  - computing power of nodes
- The problem is NP-complete
  - no exact algorithms
  - heuristics are applied

Update conflicts

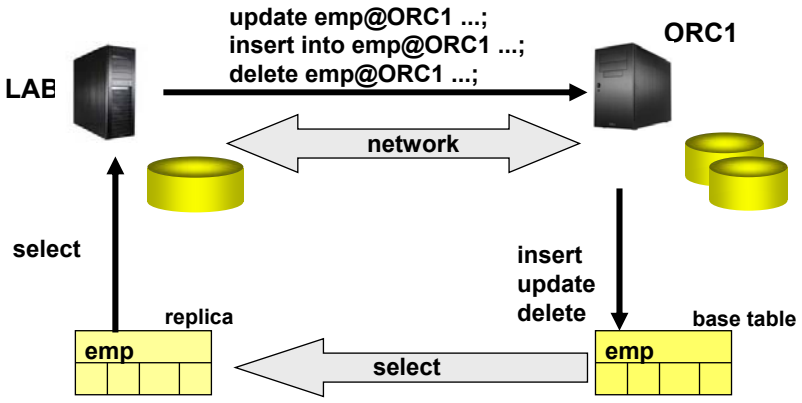
- Non unique PK values
  - central site generating PK values
  - assign each site disjoint ranges of values
  - concatenate PK value with site ID
- Conflicting updates
- Conflicting update and delete
- Conflict resolution methods
  - latest timestamp
  - earliest timestamp (rarely used)
  - site priority
  - additive (applicable to numerical values)

Clock synchronization

- Logical clock → possible duplicate values
- Central timestamp generator → bottleneck
- Clock synchronization procedure
  - every site includes its up-to-date TS value in messages sent to other sites
  - receiving node compares its TS (localTS) with the received (remoteTS) one
    - IF localTS<remoteTS THEN localTS:=remoteTS + increment_by
    - ELSE no action

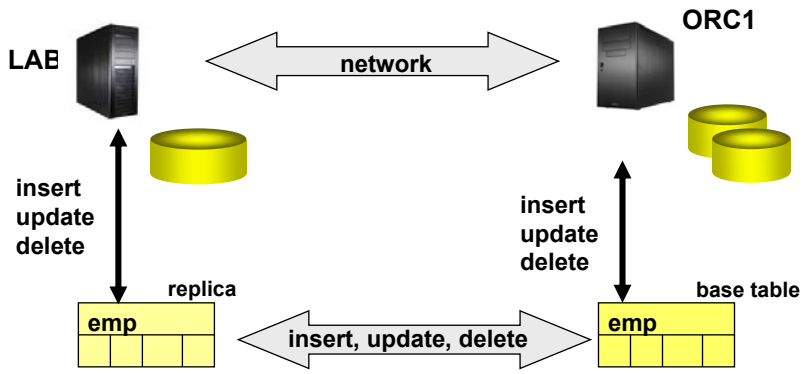
Data replication in Oracle DBMS

- standard



Data replication in Oracle DBMS

- advanced



Snapshot / materialized view

- Concept
  - a copy of a table (base table) stored in a remote DB
  - a refreshing process associated with a snapshot
- Read-only (in the standard architecture)
- Implementation
  - table + index
- Definition (SQL)
  - refreshing method (full, incremental)
    - incremental not always possible
  - refreshing moment (on demand, periodically within a defined time interval)
  - row identification for incremental refreshing (primary key, ROWID)
  - query

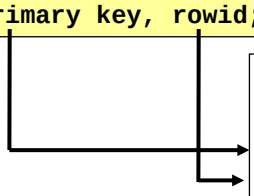
Snapshot - example

```
create snapshot AccCitiPL
refresh fast
build immediate
next sysdate+(1/24)
with primary key
as
   select * from accounts@citiPL
   where account type = 'checking';
```


Snapshot log

- Registers changes applied to the content of a snapshot's base table
- Used for incremental refreshing
- Implementation: table created and maintained by a system

```
create snapshot log on scott.emp
with primary key, rowid;
```



```
MLOG$_EMP
Name                Type
------------------- -------------
EMPNO               NUMBER(4)
M_ROW$$             VARCHAR2(255)
SNAPTIME$$          DATE
DMLTYPE$$           VARCHAR2(1)
OLD_NEW$$           VARCHAR2(1)
CHANGE_VECTOR$$     RAW(255)
```

Snapshot log - example

```
insert into emp values (1000, 'BOND', 'MANAGER', NULL,
                                '01-JAN-99', 6000, 500, 30);
update emp set comm=comm+300 where empno=7839;
delete from emp where empno=7698;
```

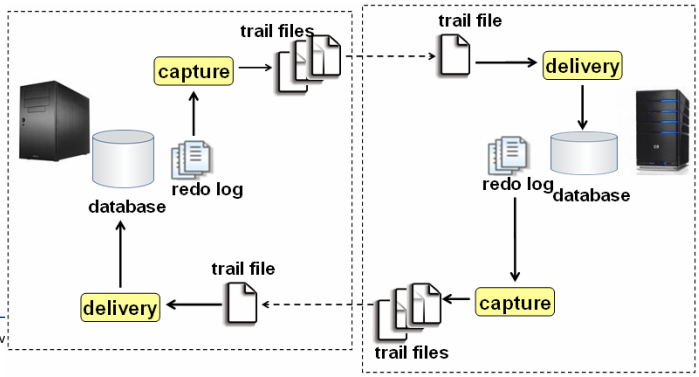
```
select * from mlog$_emp;
EMPNO M_ROW$$            SNAPTIME$ DML     OLD_  CHANGE_
                                   TYPE$$ NEW$$ VECTOR$$
----- ------------------ --------- ------ ----- --------
1000 AAAApTAACAAAAn5AAD 01-JAN-00 I      N     FEFF
7839 AAAApTAACAAAAn5AAA 01-JAN-00 U      U     8000
7698 AAAApTAACAAAAn5AAB 01-JAN-00 D      O     0000
```


Oracle GoldenGate

- "Real-time" synchronization between nodes in DDBS
  - applicable to real-time data warehousing
- Incremental replication → log sniffing
- Log-based capture from MS SQL Server, IBM DB2, MySQL
- Changes can be propagated to MS SQL Server, IBM DB2, MySQL
- Modules
  - capture
  - trail files
  - delivery

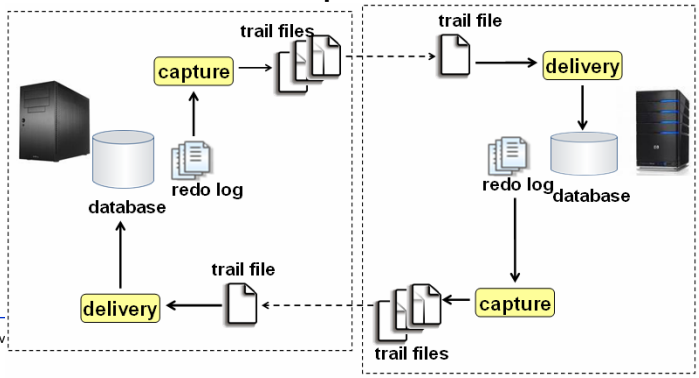
Oracle GoldenGate

- Capture: resides on the source database
  - reads inserts, updates, and deletes
  - delivers only committed transactions
- Trail files: contain the database operations for the changed data in a transportable, platform-independent data format
  - reside on the source and/or target server outside of the database



Oracle GoldenGate

- Capture module moves the captured data to the external trail file for delivery to the target
- Delivery: reads the content of a trail file and applies it to a target
  - native SQL is used for the target
  - target can also be any ODBC compatible data storage
  - the order of source transactions is preserved



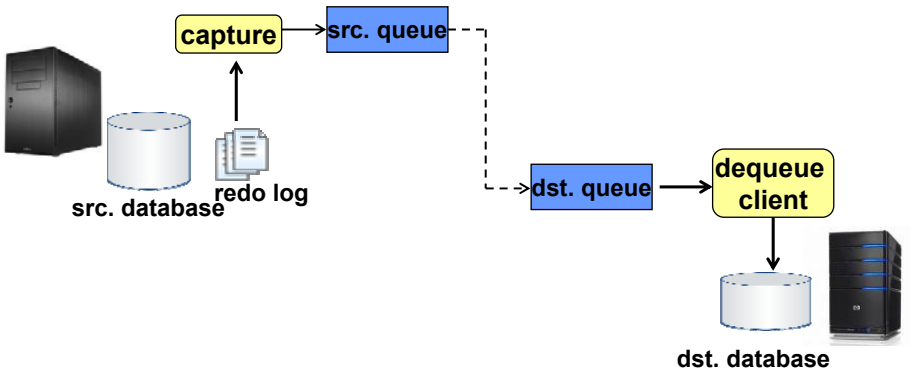
Oracle Streams

- DML and DDL captured
- Near real-time replication
- Replicated objects: entire database, a schema, tables, table fragments
- Single-master and multi-master replication, conflict resolution
- DBM_STREAMS_... packages
- Rules for capturing events

Oracle Streams

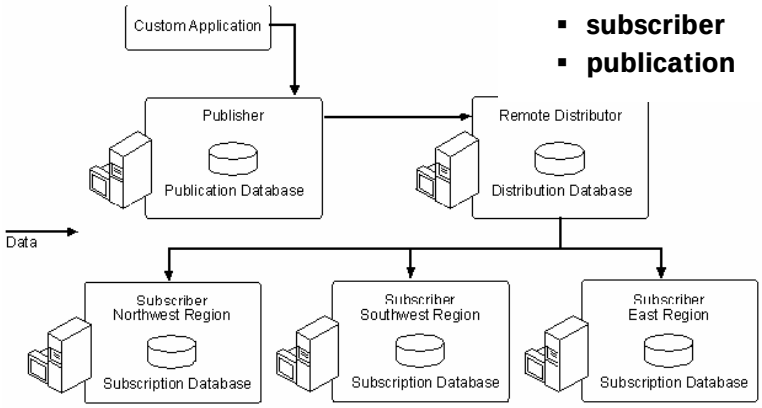
- Process Capture: detects database events DDL and DML → redo log analysis
  - rules used by a capture process determine which changes it captures
  - captured changes (messages) are stored in a queue
  - modes of capturing
    - from online redo log
    - from archived redo log
- Dequeue Client
  - can be Process Apply or a user application
  - rules determine which messages are dequeued
  - multiple destination databases can read from the same queue

Oracle Streams



Data replication in SQL Server

- Actors (nodes)
  - publisher
  - distributor
  - subscriber
  - publication



Replication model

- Publisher
  - a server providing data for replication
  - replicated data are organized in publications
  - a publisher can provide multiple publications
- Distributor
  - a server storing data for replication
  - responsible for replication
- Publication
  - a unit of replication
  - contains at least one article

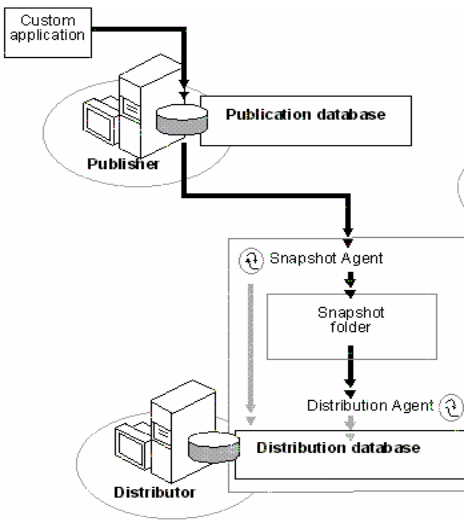
Replication model

- Article
  - database object to be replicated
  - can be
    - a table
    - a subset of table's columns
    - a subset of table's rows
- Subscriber
  - a node receiving publications
- Subscription
  - contains publications
  - defines replication schedule
  - a set of subscribers
  - push / pull subscription

Replication model

- Agents manage various types of replications
  - SQL Server Agent
    - managing other agents and processes
    - monitoring errors
  - Snapshot Agent
  - Log Reader Agent
  - Distribution Agent
  - Merge Agent
- Replication types
  - snapshot (full)
  - transactional (incremental)
  - merge (incremental)
    - modification to replica tables are propagated to a master table

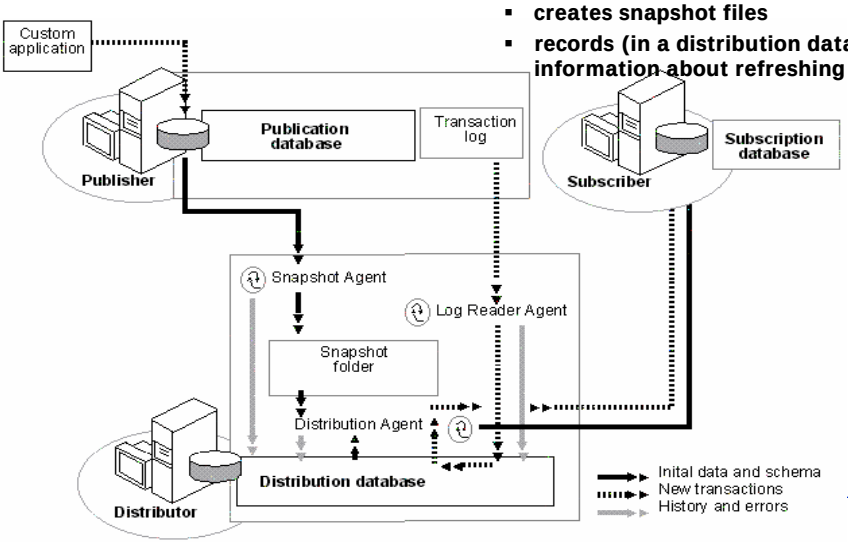
Snapshot replication



- Snapshot Agent
  - creates snapshot files
  - records (in a distribution database) information about refreshing
- Distribution Agent
  - sends a snapshot to a subscriber

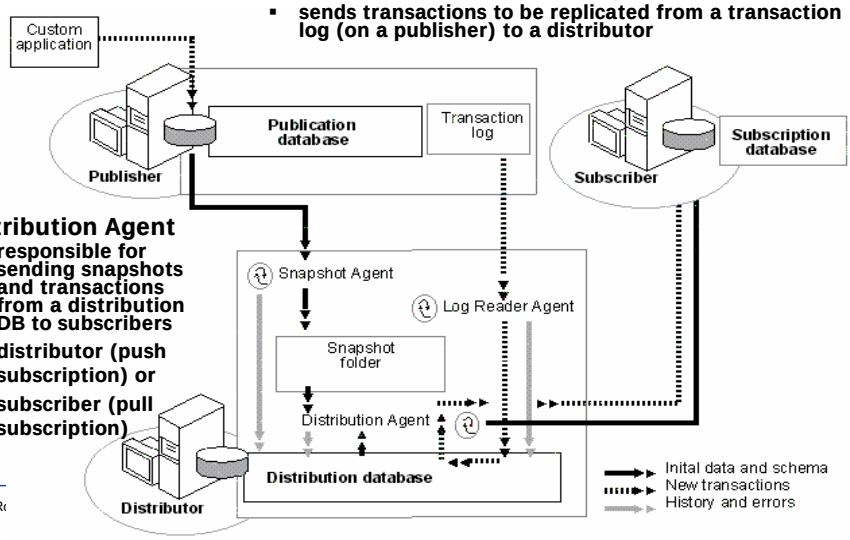
Transactional replication

- **Snapshot Agent**
  - **creates snapshot files**
  - **records (in a distribution database) information about refreshing**



Transactional replication

- **Log Reader Agent**
  - **monitors a transaction log of a publisher**
  - **sends transactions to be replicated from a transaction log (on a publisher) to a distributor**



- **Distribution Agent**
  - **responsible for sending snapshots and transactions from a distribution DB to subscribers**
  - **distributor (push subscription) or**
  - **subscriber (pull subscription)**

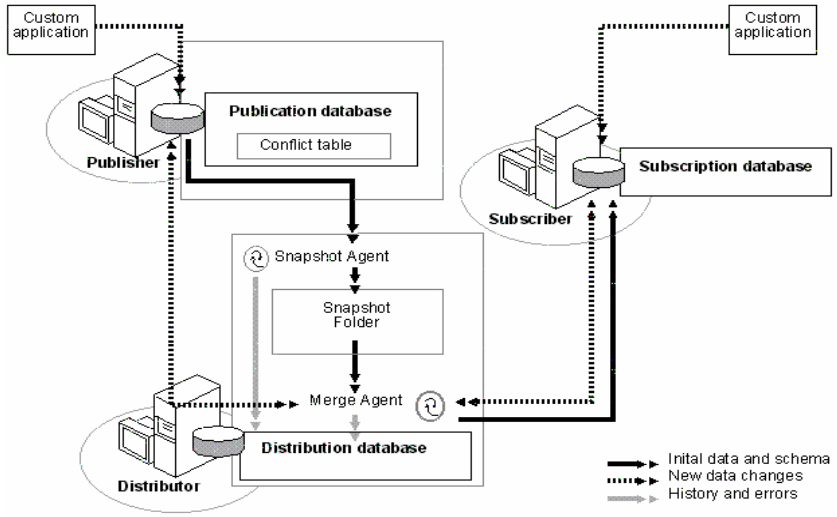
Merge replication

- System tracks changes in a publisher and subscribers
- Multiple replicas of the same base table can be updated
- Conflicts may occur during updates merging
  - merge agent responsible for selecting the right version of data

Merge replication

- Snapshot Agent
  - creates initial snapshots
  - stores snapshot files on a distributor
  - registers information about synchronization and stores them on a distributor
  - creates system tables, procedures, and triggers for this type of replication
- Merge Agent
  - sends an initial snapshot to subscribers
  - merges updates from subscribers
  - resolves conflicts

Merge replication

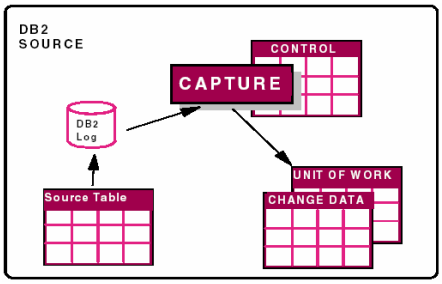


Data replication in IBM DB2

- **Features**
  - **incremental**
  - **one-way (replica is read-only)**
  - **refreshing**
    - **automatic**
    - **automatic synchronous**
    - **event based**
- **Processes**
  - **CAPTURE – detecting changes in base tables**
  - **APPLY – applying detected changes to replicas**
  - **MONITOR – manages and monitors a replication**

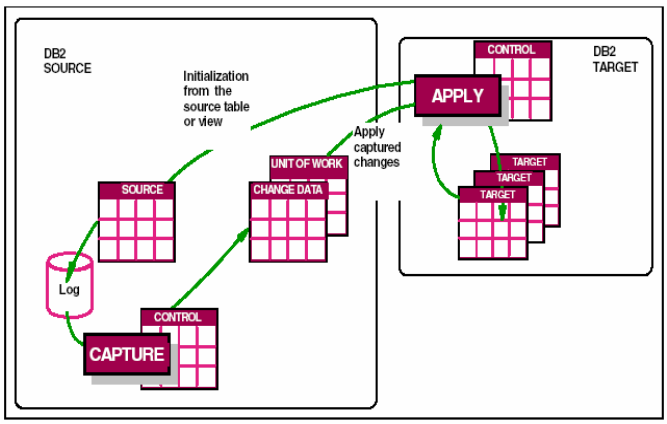
CAPTURE

- Detects and retrieves updates to the content of a base table from a transactional log
- Each base table has associated table CHANGE DATA created by a system
- Table UNIT OF WORK stores information on committed transactions



APPLY

- Applies changes prepared by CAPTURE to replicas



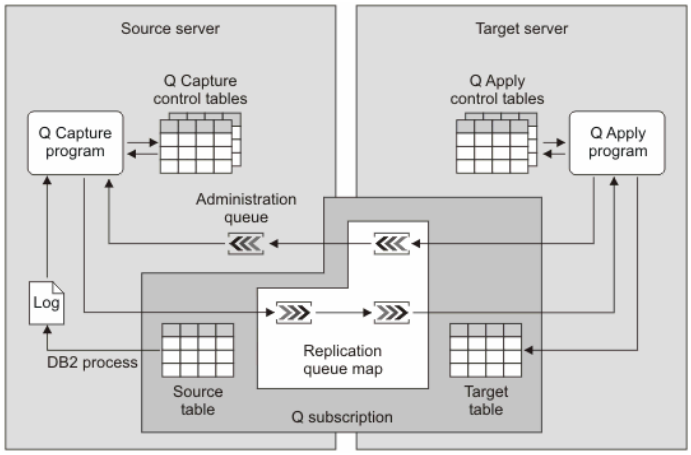
IBM Queue replication

- Connections between databases can be off
- Databases communicate by means of queues
- Incremental
- Replication of vertical and horizontal fragments of source tables
- One-way
  - one source, multiple replicas
  - possible data filtering
- Two-ways
  - for 2 servers only (primary and secondary)
  - one server configured as a consistent winner
- Peer-to-peer
  - multiple servers with the same priority
  - conflicts resolved based on timestamps

IBM Queue replication

- Replication to systems
  - DB2
  - Informix (target only)
  - MS SQL Server (target only)
  - Oracle (target only)
  - Sybase (target only)
- IBM InfoSphere Replication Server
- Committed data are sent to the queuing system → message queue
- At the target messages are read from the queue and converted into transactions

IBM Queue replication



Introduction to Q replication at
http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=/com.ibm.swg.im.iis.repl.qtutorial.doc/topics/iiyrqtutabstr1.html

Replication in open source systems

- Daffodil Replicator
  - JDBC drivers for data sources
  - publication-subscription replication model
    - publication: the set of tables + filtering
  - replication modes
    - snapshot
    - merge
    - pull
    - push

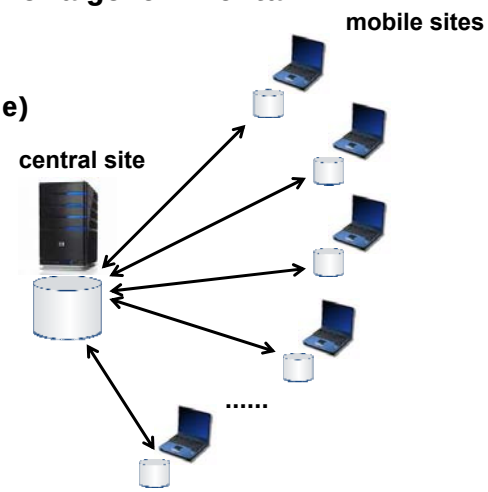
Replication in open source systems

Source	Destination	Result
PostgreSQL	Firebird	snapshot
Firebird	PostgreSQL	failed
PostgreSQL	MySQL	snapshot
MySQL	PostgreSQL	failed
MySQL	Firebird	snapshot, pull, push, merge
Firebird	MySQL	snapshot, pull, push, merge

- **Problems**
  - **errors in scripts that generated DB objects (tables, triggers) managing replication**

Case study

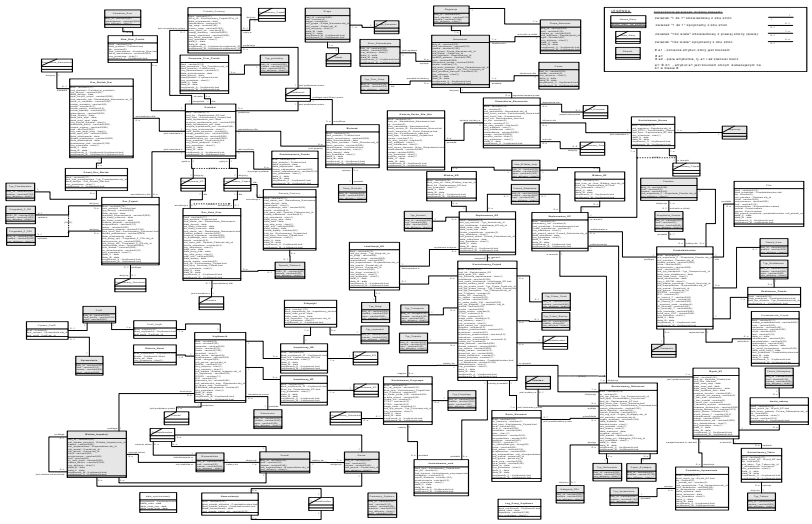
- **Distributed and mobile DBS for a governmental institution**
- **Central site**
  - **Central DBMS server (Oracle)**
  - **Web server (Oracle)**
  - **Java applications**
- **Mobile sites**
  - **300 notebooks**
  - **Oracle Lite DBMS**
  - **Tomcat**
  - **Java applications**



Case study

- Challenges
  - data synchronization central DBMS ←→ mobile DBMS
    - on demand, incremental, 2-way replication
    - user (context) aware replication - not all data replicated to all mobile DBs
- 82 central tables
  - 47 dictionary tables replicated from central DBMS to mobile DBMS
  - 35 tables replicated in both directions
    - user-implemented log-based replication
    - PL/SQL user-implemented packages

Case study



Case study

- Data exchange with external systems
  - XML files
  - user-implemented incremental update
  - duplicated entries in the input file → duplicate elimination