Dbvisit
THE SMART ALTERNATIVE

# The 3 fundamental principles of Oracle replication

Presenter: Arjen Visser, Founder and CTO of Dbvisit Software



# Introduction

Arjen Visser @dbvisit
Founder and CTO of Dbvisit Software Limited, Auckland, New Zealand
Technical Entrepreneur with passion for Oracle Database

Past Experience:

- DBA
- Unix admin/project manager
- Datawarehouse developer/programmer
- Speaker at OOW, NZOUG, CLOUG, RMOUG, Collaborate, DOAG

# Dbvisit Software

Dbvisit
THE SMART ALTERNATIVE

- Dedicated software development company, specialize in Oracle Replication
- Offices in US and Europe. HQ in Auckland, New Zealand
- World wide leader in DR solutions for Oracle Standard Edition
- Product Engineers with "real world" DBA Experience
- Two Oracle 11g Certified Masters
- In business 6 years, Growing rapidly

amazon web services | Partner Network
TECHNOLOGY PARTNER

Deloitte.
Technology Fast500
Asia Pacific 2012 Winner

technology fast 50 | Deloitte.

ORACLE Gold Partner

# Trusted in over 80+ countries . . .


. . . by 600 companies.

# Our portfolio

Dbvisit
THE SMART ALTERNATIVE


Data Guard alternative


Golden Gate alternative


Active Data Guard alternative

# Objective

Understanding the fundamental principles and considerations of Oracle logical replication

# Agenda

- The difference between physical and logical replication
- How does logical replication work
- 3 replication principles and how they apply (including demos)
- Considerations with logical based replication

# Two Oracle Replication Types

Dbvisit
THE SMART ALTERNATIVE

## Physical Replication

- "One to one" copy of the primary database in recovery state
- Use redo apply to keep up to date
- 100% binary copy, structure and data
- Referred to as a standby database
- **Best suited for DR**

## Logical Replication

- Independent 2nd database kept in sync by replication mechanism
- Uses SQL statements to keep database up to date
- Subset of data is replicated
- Cross version, cross platform
- Separate physical database structure
- **Best suited for information sharing, migrations, real-time reporting etc**



# Logical replication options

1. Trigger based
   - Changes to source database are required
   - Performance impact
   - A lot more maintenance
2. Redo log mining (preferred Oracle solution)
   - No changes to source database
   - Lower impact to source environment
   - Less maintenance
   - DDL changes can be replicated

# Redo log - Redo gets it first

Changes written to database in order:

1. Written to redo log buffer
2. Written from redo log buffer to redo log (after commit, 3 seconds or 1/3 full)
3. Written to database files

Redo will include:

1. Insert, Update, Delete, Merge
2. Select for Update
3. DDL
4. No Select

# How does redo log mining logical replication work

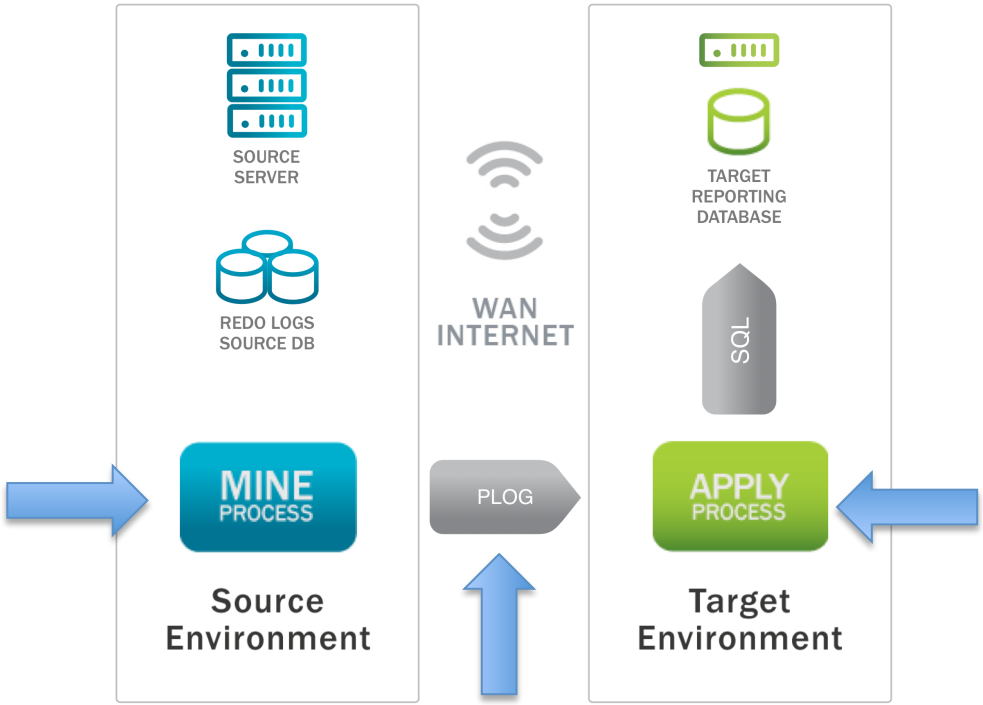
Dbvisit
THE SMART ALTERNATIVE

1. Redo logs are mined for LCR* as Oracle is writing to the logs (real time)
2. LCR are pieced together in correct order (ie row chaining, row migration, RAC)
3. Filtered LCR are written to a file or a queue
4. Transferred to target server
5. Translated to SQL statements
6. Run against target database

* Logical Change Record - describes a change to a row



# Typical logical replication flow




## Mine

Mines the redo logs and converts into a parsed log

## PLOG

Parsed logs – binary files specific to solution

Platform independent

## Apply

Converts parsed log data into target DB native SQL

# Redo log mining solutions

1. Standalone products (own mining engine)
   - Golden Gate
   - Dbvisit Replicate
2. LogMiner - standard with database
3. Streams - Oracle EE (may be un-supported in future)

# 3 principles of Oracle redo log mining replication

Dbvisit
THE SMART ALTERNATIVE

1. Conversion process
   - All set based SQL operations are converted to row-by-row SQL changes
2. Identification
   - For the data in the table to be replicated, each row in the table must be uniquely identified
3. Conflicts
   - Conflicts warn of data divergence and lost updates



# Principle 1 - Conversion process

Dbvisit
THE SMART ALTERNATIVE

Set based SQL operations on the source database are converted to row-by-row SQL changes on the target database.

Why?

- This is the way that Oracle writes to the Redo
- Redo does not contains SQL, it has to be rebuild from LCRs

Observations:

- SQL is not the same on source as on target
- True for all logical replication based solutions
- Each SQL on target only affects 1 row



# 1. Logical replication principle

| Source | Target |
| --- | --- |
| `update PRICES set PRICE = PRICE - (PRICE * .10) where PRODUCT_CAT = 'OLD_STOCK';` | `update PRICES set PRICE = 10 where PROD_ID = 101;`<br>`update PRICES set PRICE = 23 where PROD_ID = 102;` |

## Observations:

1. Source SQL updates 2 rows, then 2 individual update statements are produced
2. PK has been added to the WHERE to ensure row-by-row
3. Price formula has been replaced by hardcoded value

# 1. Logical replication principle

## Question:

What if the table to be replicated does not have a primary key? Does that mean we cannot replicate this table using logical based replication?

## Answer:

Depends on logical replication principle - 2

# Principle 2 - Identification

Dbvisit
THE SMART ALTERNATIVE

For the data in the table to be successfully replicated, each row in the source table must be uniquely identified

To uniquely identify each row, ONE of the following must be true:

1. Primary key
2. Unique key
3. All data in columns in row must produce a unique result

If none are true, then table replication may cause conflicts



# 2. Logical replication principle

## Example

```
CREATE TABLE SALES
    (PROD_ID NUMBER PRIMARY KEY,
     CUST_ID NUMBER NOT NULL,
     QUANTITY_SOLD NUMBER(10,2) NOT NULL,
     AMOUNT_SOLD NUMBER(10,2) NOT NULL,
     AMOUNT_RECEIVED NUMBER(10,2),
     SALES_STATUS VARCHAR2(10) NOT NULL
    );
```

# 2. Logical replication principle

Source statement:

```
update SALES set AMOUNT_RECEIVED = 120
where SALES_STATUS = 'OVERDUE'
```

(old value AMOUNT_RECEIVED is NULL)

Target statement:

- Primary key (prod_id)

```
update SALES set AMOUNT_RECEIVED = 120
where PROD_ID = 101
and AMOUNT_RECEIVED is NULL
```

PK in where clause

Old value of updated column in where clause

# 2. Logical replication principle

- Unique key (prod_id, cust_id)

```
update SALES set AMOUNT_RECEIVED = 120
where PROD_ID = 101
and CUST_ID = 201
and AMOUNT_RECEIVED is NULL
```

UK in where clause

Old value of updated column in where clause

# 2. Logical replication principle

- No primary or unique keys

```
update SALES set AMOUNT_RECEIVED = 120
where SALES_STATUS = 'OVERDUE'
and PROD_ID = 101
and CUST_ID = 201
and QUANTITY_SOLD = 233
and AMOUNT_SOLD = 1299
and AMOUNT_RECEIVED is NULL
```

All columns in where clause

Old value of updated column in where clause

If all columns in the where clause does not uniquely identify the row, then the table cannot be guaranteed to be replicated

Why? Because of principle - 3

# Principle 3 - Conflicts

Conflicts warn of data divergence and lost updates

Conflict = potential for the data between source and target to be out of sync – data divergence.

Conflict when the SQL on the target (apply) affects

- Zero rows
- More than one row (>1)
- Oracle errors (Unique key and Foreign key violations)

# 3. Logical replication principle

Example of conflict

- update statement updates 0 rows
- delete statement deletes 0 rows
- update statement updates 2 or more rows
- delete statement deletes 2 or more rows

Why are these conflicts? Because otherwise it breaks principle 1 Set based SQL operation on the source database are converted to row-by-row SQL changes on the target database

# 3. Logical replication principle

- Update statement replicated from source

```
update SALES set AMOUNT_RECEIVED = 120
where SALES_STATUS = 'OVERDUE'
and AMOUNT_RECEIVED is NULL
and PROD_ID = 101
```

- Meantime someone on target has updated row

```
update SALES set AMOUNT_RECEIVED = 60
where SALES_STATUS = 'OVERDUE'
and PROD_ID = 101
```

- Result:
  - Conflict – 0 rows updated
  - Potential to lose update (amount_received = 60)

# Demos of replication principles

1. Real world example of lost update
2. Delete conflict with non unique data

# Review - 3 principles of Oracle logical replication

1. Conversion process
   - All set based SQL operations are converted to row-by-row SQL changes
2. Identification
   - For the data in the table to be replicated, each row in the table must be uniquely identified
3. Conflicts
   - Conflicts warn of data divergence and lost updates

More info: http://bit.ly/N1LII4

# Considerations with logical based replication

1. Effectively manage conflicts
2. Data instantiation
3. Updating Primary Keys
4. Triggers and cascading constraints
5. Sequences
6. Commit strategies
7. Generic replication principles

# Considerations with logical based replication

1. Effectively manage conflicts
    1. Conflict detection

       when there is a possibility of data divergence, a notification or alert is triggered.
    2. Conflict resolution

       when a conflict occurs, enough information about the conflict is known so that the conflict can be resolved
    3. Conflict handling

       set predefined rules to automatically determine what to do when conflict occurs (includes setting PL/SQL business rules)

# Considerations with logical based replication

## 2. Data instantiation

Before replication can start, data needs to be in sync between source and target. Ensure:

- All data is captured including in-flight data
- All required indexes
- PL/SQL

Use

1. Datapump (export/import)
2. Rman duplicate
3. Standby database (resetlogs)

# Considerations with logical based replication

Dbvisit
THE SMART ALTERNATIVE

## 3. Updating Primary Keys

Bad idea, and adds further complications when replicating

- Do not use batched SQL

```
SQL> update SCOTT.DEPT set DEPTNO = 50 - DEPTNO where
DEPTNO in (20,30);
```

- Use single update statements with temporary values

```
SQL> update SCOTT."DEPT" set DEPTNO = 29 where DEPTNO = 20;
```

See http://bit.ly/YZi2Bs for more info



# Considerations with logical based replication

Dbvisit
THE SMART ALTERNATIVE

4. Triggers and cascading constraints

Beware of side effects

These are recursive SQL/changes and they also get replicated at row level. So if you have such trigger/FK, you end up applying the same change twice (and create conflicts).

Turn off or disable triggers at the target for data that is replicated.

Some products can deal with this



# Considerations with logical based replication

5. Sequences

- Replication does not update target Oracle sequences
- Value inserted by sequence is replicated

End result

- Primary key values will be the same on both source and target
- Sequences will not be the same

Ensure strategy in place for Active-active or migrations

# Considerations with logical based replication

6. Commit strategies
    1. Optimistic commit (do not wait for commit before mining and applying)
    2. Wait for commit before applying (do not wait for commit before mining)
    3. Wait for commit before mining and applying

Examples of products with above strategies:

1. Dbvisit Replicate
2. Streams
3. Golden Gate

# Considerations with logical based replication

Dbvisit
THE SMART ALTERNATIVE

7. Generic replication considerations
    1. Replicated data is same order as source (based on SCN)
    2. Transaction integrity is maintained (replication is serial)
    3. Autonomous transactions are maintained (commit & rollback)
    4. Conflicts pauses the whole replication to ensure integrity



# Logical replication

- Very powerful but also complex
- Opens up many possibilities, be creative
- Many use cases
- Make sure you understand the fundamentals

**REPORTING & MIGRATIONS**

**DATA DISTRIBUTION**
Example: replicating from head office to remote offices, e.g. pricing information.

**ACTIVE/ACTIVE**
Example: applications and data that continuously support both regional requirements and global operations.

**REAL-TIME BI & AUDITING**

**CASCADING FLOW**

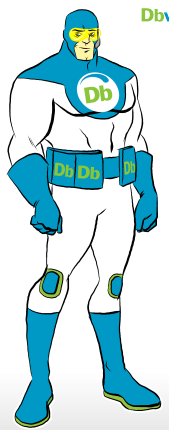
# Come by our booth for Dbman tattoos!

Dbvisit
THE SMART ALTERNATIVE

Dbvisit standby

Dbvisit replicate

Dbvisit reporting




@dbvisit
blog.dbvisit.com

Join our Auckland Oracle DBA meetup - on meetup.com