Oracle’s Change Data Capture (CDC)

Boris Knizhnik

BIK Information Services, Inc.

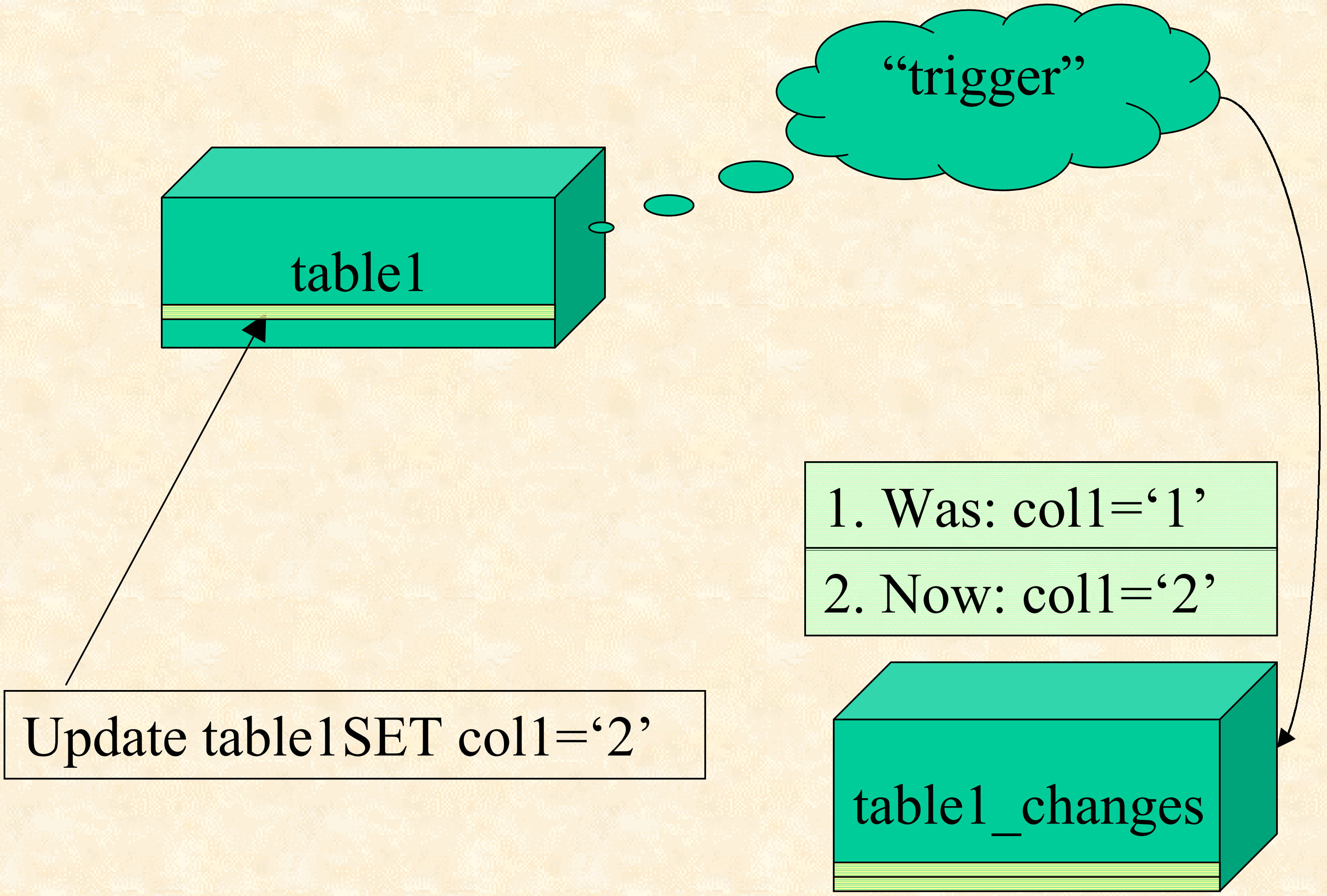
borisk@bikinfo.com

NYOUG 12/11/2003

What is Change Data Capture?

- ❑ Tool to help manage data changes
- ❑ NOT a data warehousing solution
- ❑ Can be used as a part of data warehousing solution
- ❑ Doesn’t require any changes to existing database design

Basic concept



“trigger”

table1

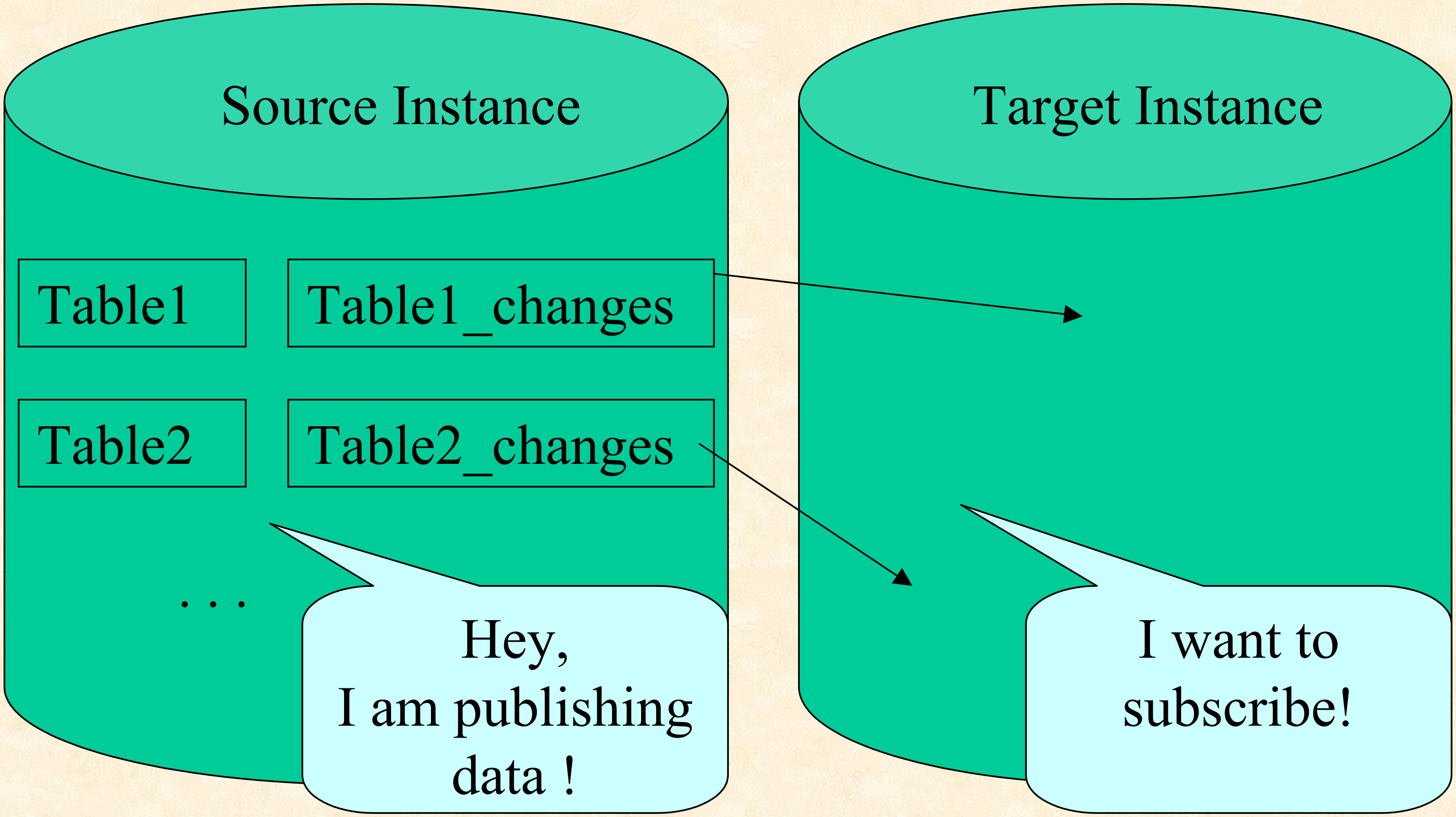
1. Was: col1=‘1’

2. Now: col1=‘2’

Update table1SET col1=‘2’

table1_changes

Basic concept (cont.)



Source Instance

Target Instance

Table1

Table1_changes

Table2

Table2_changes

. . .

Hey,
I am publishing
data !

I want to
subscribe!

Preparations

- ❑ Make sure you know what tables you will use in CDC process.
- ❑ If tables are still under development – use the utility package to build the list of columns on the fly.
- ❑ Prepare two accounts – publisher and subscriber.

Setting up Publisher

```
connect system/manager@whatever
GRANT EXECUTE_CATALOG_ROLE to
boris_publisher;
GRANT SELECT_CATALOG_ROLE to
boris_publisher;
```

```
connect scott/tiger@whatever
GRANT SELECT on emp to boris_publisher;
GRANT SELECT on DEPT to
boris_publisher;
```

Create Change Tables

```
DBMS_LOGMNR_CDC_PUBLISH.CREATE_CHANGE_TABLE (
 CHANGE_SET_NAME => 'SYNC_SET‘
, CAPTURE_VALUES => 'both'
, RS_ID => 'y', ROW_ID => 'n', USER_ID => 'y', TIMESTAMP => 'y'
, OBJECT_ID => 'n‘, OPTIONS_STRING => null
, SOURCE_COLMAP => 'y', TARGET_COLMAP => 'y'
, OWNER => 'boris_publisher', SOURCE_SCHEMA => 'scott'
, SOURCE_TABLE => 'emp'
, CHANGE_TABLE_NAME => 'cdc_emp'
, COLUMN_TYPE_LIST =>
'comm number(7,2),deptno number(2),empno number(4) '
|| ',ename varchar2(10),hiredate date,job varchar2(9) '
|| ',mgr number(4), sal number(7,2)'
);
```

Inside Change Table

```
 Name                          Type
------------------             -------------
OPERATION$                     CHAR(2)
CSCN$                          NUMBER
COMMIT_TIMESTAMP$              DATE
RSID$                          NUMBER
USERNAME$                      VARCHAR2(30)
TIMESTAMP$                     DATE
SOURCE_COLMAP$                 RAW(128)
TARGET_COLMAP$                 RAW(128)
COMM                           NUMBER(7,2)
DEPTNO                         NUMBER(2)
...
```

Columns from the original table

Another publishing scenario

- Same table may be published more than once
- Each change table for the same source table may contain a different number of columns

Setting up Subscriber

```
connect scott/tiger@whatever
GRANT SELECT ON emp TO boris_subscriber;
GRANT SELECT ON dept TO boris_subscriber;
```

```
connect boris_publisher/boris_publisher
GRANT SELECT ON cdc_emp TO boris_subscriber;
GRANT SELECT ON cdc_dept TO boris_subscriber;
```

Creating a Subscription

```
DECLARE vHandle NUMBER;

v_subscription_description := 'scott -> Datawarehouse';
. . .

DBMS_CDC_SUBSCRIBE.GET_SUBSCRIPTION_HANDLE(
          CHANGE_SET => 'SYNC_SET',
          DESCRIPTION => v_subscription_description,
          SUBSCRIPTION_HANDLE => vHandle
);
```

Result

Creating a Subscription (cont.)

```
DECLARE col_names VARCHAR2(2000);
v_source_schema VARCHAR2(20) := 'SCOTT';
v_source_table VARCHAR2(31);
```

```
v_source_table := 'EMP';
col_names := 'comm,deptno,empno,ename,hiredate,job,mgr,sal ';
DBMS_LOGMNR_CDC_SUBSCRIBE.SUBSCRIBE (vHandle,
          v_source_schema, v_source_table, col_names);
```

```
v_source_table := ' DEPT';
col_names := ' deptno,dname,loc ';
DBMS_LOGMNR_CDC_SUBSCRIBE.SUBSCRIBE (vHandle,
          v_source_schema, v_source_table, col_names);
```

. . .

Activate a Subscription

```
DECLARE v_subscription_description VARCHAR2(30) :=
                          'scott -> Datawarehouse';

-- Get the handle
    SELECT handle INTO vHandle FROM all_subscriptions
            WHERE description = v_subscription_description;
-- Activate the subscription
DBMS_CDC_SUBSCRIBE.ACTIVATE_SUBSCRIPTION(vHandle);

-- Extend the subscription window
DBMS_CDC_SUBSCRIBE.EXTEND_WINDOW(
        SUBSCRIPTION_HANDLE=>vHandle);
```

Logistical problem

- Processing data in a change table takes some time.
- In the mean time new records could have been stored in this change table.
- After you have processed the records, the next time your processing program kicks in, you may have a few more records in those tables.
- How are you going to tell the old processed records from the new ones?

Solution: Extend_window

Extending Window

-- get the handle

```
SELECT handle INTO vHandle FROM all_subscriptions
                WHERE description = v_subscription_description;
```

```
DBMS_CDC_SUBSCRIBE.EXTEND_WINDOW(
           SUBSCRIPTION_HANDLE=>vHandle);
```

Cyclical Part

- ❑ Publisher created change tables and is constantly collecting change records.
- ❑ Subscriber specified which of these tables she is interested in.

- ❑ We are ready for a cyclical part of processing collected records
- ❑ Reading change tables directly is not recommended by Oracle, because the tables are not stable.
- ❑ The number of records keeps growing while your data warehouse process reads these records.

The solution is to create views that give you a fixed set of records for each underlying change table.
After your data warehouse script finishes processing records, you may drop this view.

Extending Window and Creating CDC Views

```
connect boris_subscriber/boris_subscriber@whatever
. . .
-- Get the handle
SELECT handle INTO vHandle FROM all_subscriptions
    WHERE description = v_subscription_description;
-- Extend the window for subscription
DBMS_CDC_SUBSCRIBE.EXTEND_WINDOW(
          SUBSCRIPTION_HANDLE=>vHandle);
-- Create CDC View (for each table)
v_cdc_table := 'CDC_'||v_source_table;
DBMS_CDC_SUBSCRIBE.PREPARE_SUBSCRIBER_VIEW(
          SUBSCRIPTION_HANDLE=> vHandle
          ,SOURCE_SCHEMA => v_source_schema
          ,SOURCE_TABLE => v_source_table
          ,VIEW_NAME => our_view_name);
```

Result variable

Extending Windows and Creating CDC Views (cont.)

-- Drop the previous synonym

```
vSQL := 'DROP SYNONYM ' || v_cdc_view_name;
EXECUTE IMMEDIATE vSQL;
```

-- Create a private synonym to point to the view for each table:

```
v_cdc_view_name:=v_cdc_table|| '_vw';
vSQL := 'CREATE SYNONYM ' || v_cdc_view_name ||
        ' FOR '|| our_view_name;
EXECUTE IMMEDIATE vSQL;
```

CDC Views and Synonyms

Subscriber view 'CDC#CV$8757846' was successfully created for table SCOTT.DEPT

Private synonym 'CDC_DEPT_vw' for view 'CDC#CV$8757846' was successfully created.

Subscriber view 'CDC#CV$8757848' was successfully created for table SCOTT.EMP

Private synonym 'CDC_EMP_vw' for view 'CDC#CV$8757848' was successfully created.

CDC Views and Synonyms (cont.)

```
CREATE OR REPLACE VIEW CDC#CV$8757846 ( OPERATION$,
CSCN$, COMMIT_TIMESTAMP$, TIMESTAMP$, USERNAME$,
TARGET_COLMAP$, SOURCE_COLMAP$, RSID$, DEPTNO,
DNAME, LOC
) AS SELECT
OPERATION$, CSCN$, COMMIT_TIMESTAMP$,   TIMESTAMP$,
USERNAME$, TARGET_COLMAP$, SOURCE_COLMAP$, RSID$,
"DEPTNO", "DNAME", "LOC"
FROM "BORIS_PUBLISHER"."CDC_DEPT"
WHERE CSCN$ >= 40802127 AND CSCN$ <= 41013754
WITH READ ONLY
```

Processing Change Records

```
SELECT * FROM CDC_DEPT_vw
ORDER BY
```

```
CSCN$
, COMMIT_TIMESTAMP$, TIMESTAMP$
, OPERATION$ Desc
```

Note 1: Don’t forget to specify ‘order’ clause!

Note 2: Watch for batches that update millions of records!

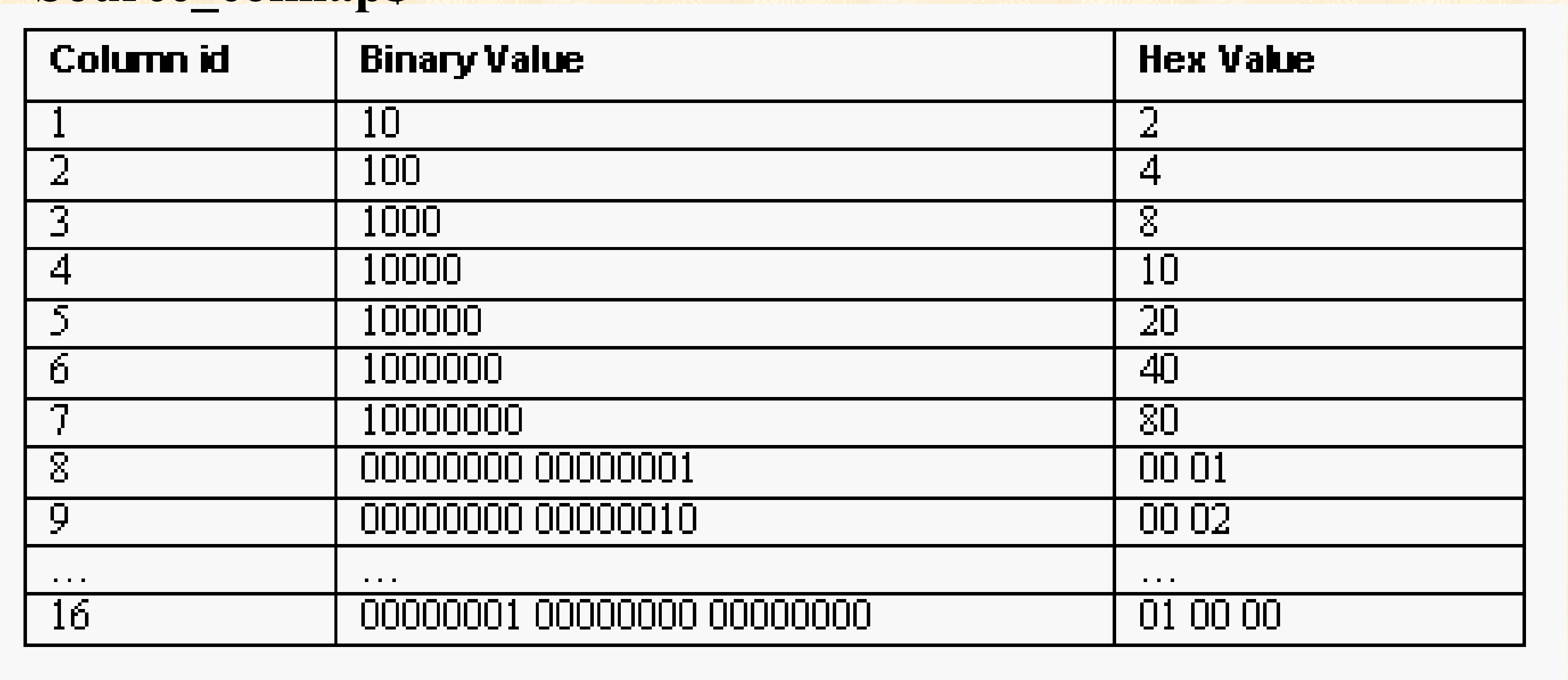
Initial Load ?

Consider creating views such as this:

```
CREATE VIEW CDC_EMP_VW AS
SELECT 'I' operation$, 1 cscn$
, SYSDATE commit_timestamp$
, 1 rsid$
, 'initial_load' username$,
 SYSDATE timestamp$
, HEXTORAW('FEFFFFFF)' SOURCE_COLMAP$
, HEXTORAW('FEFFFFFF') TARGET_COLMAP$
, t.* FROM emp t;
```

What columns were changed

Source_colmap$



Column id	Binary Value	Hex Value
1	10	2
2	100	4
3	1000	8
4	10000	10
5	100000	20
6	1000000	40
7	10000000	80
8	00000000 00000001	00 01
9	00000000 00000010	00 02
…	…	…
16	00000001 00000000 00000000	01 00 00

Apparently Oracle’s inner presentation of the values is as a set of binary words (two bytes). For historical reasons, these are usually reversed in memory presentation. The least significant byte comes first and the most significant byte follows.

What columns were changed (cont.)

- ❑ Learning what columns have been changed may be important.
- ❑ Using SOURCE_COLMAP$ may not give you the correct results since Oracle does not check whether or not the values really changed.
- ❑ It grabs columns that were mentioned in the UPDATE statement even if this statement is assigning the same values back.

Dropping CDC Views

```
connect boris_subscriber/boris_subscriber@whatever
-- Get the handle
SELECT handle INTO vHandle FROM all_subscriptions
     WHERE description = v_subscription_description;
-- Drop the synonym
 vSQL := 'DROP SYNONYM ' || v_cdc_view_name;
EXECUTE IMMEDIATE vSQL;
-- Drop the subscriber view(s) – for all tables
v_source_table := ' emp ';
DBMS_CDC_SUBSCRIBE.DROP_SUBSCRIBER_VIEW(
SUBSCRIPTION_HANDLE=> vHandle
          ,SOURCE_SCHEMA => v_source_schema
          ,SOURCE_TABLE => v_source_table);
```

```
Subscriber View for table 'CDC_DEPT' was dropped. Handle # 86
Subscriber View for table 'CDC_EMP'   was dropped. Handle # 86
```

Purge the subscription window

```
-- Get the handle
SELECT handle INTO vHandle FROM all_subscriptions
    WHERE description = v_subscription_description;

-- Purge window
```

```
DBMS_CDC_SUBSCRIBE.PURGE_WINDOW(
         SUBSCRIPTION_HANDLE=> vHandle);
```

Subscriber Window for subscription 'scott -> Datawarehouse' was successfully purged

Practical Advice

A slightly different sequence of steps is recommended for a production environment:

Step 1 – drop the CDC views (this will fail the first time, since there are none)
Step 2 – purge the CDC window (this will also fail the first time)
Step 3 – extend the windows, create CDC views, create synonyms
Step 4 – process updates

This sequence leaves your CDC views intact between runs and you can do the research what went wrong between runs.

Advice (Cont.)

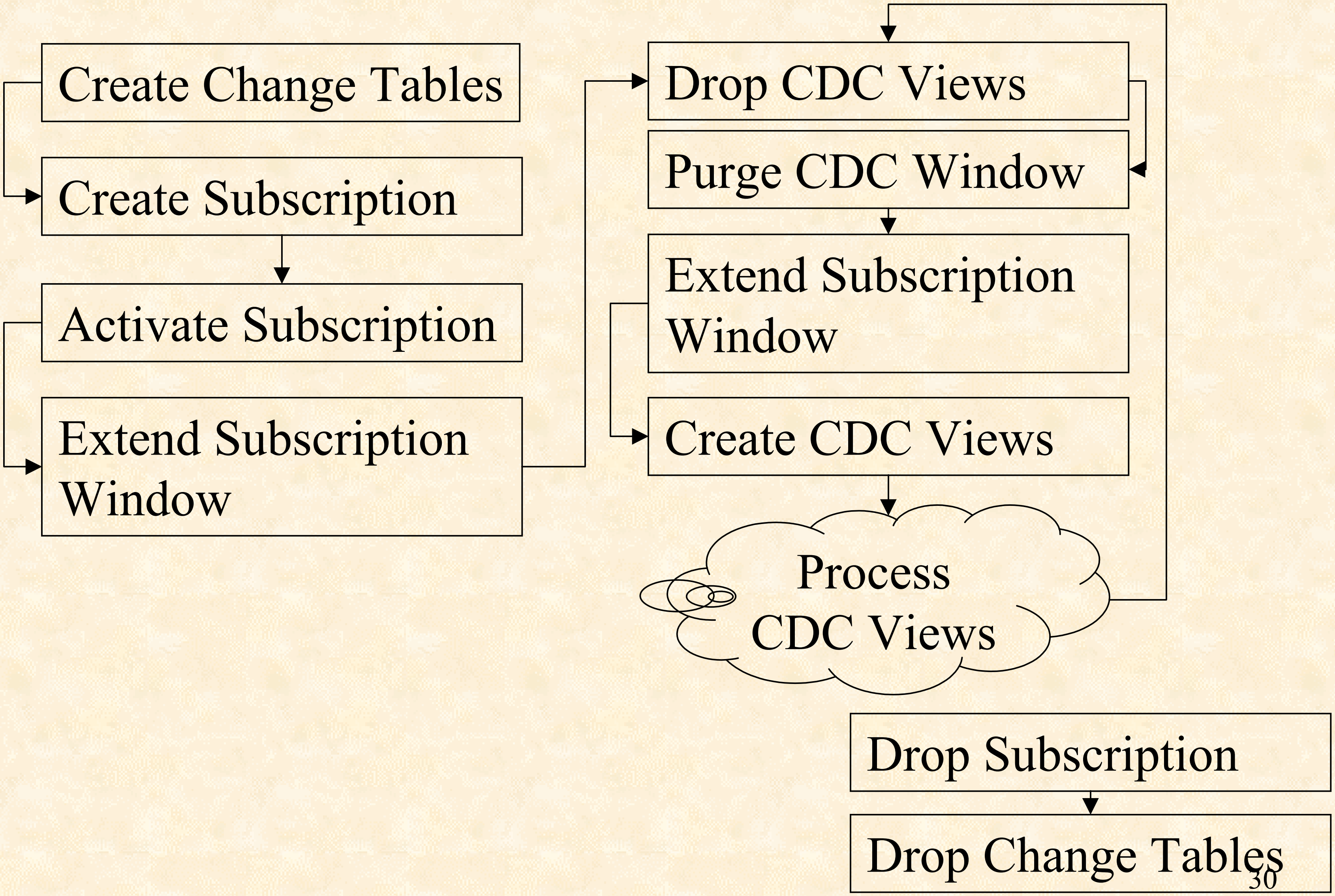
- If your source database is really on another instance, your update process will be the one with a lot of @db_link tables.
- It is a good idea to design the update process in such a way that it could be applied again without causing problems.
  - You may want to treat Inserts as Updates if the key already exists in a target database or Deletes will not really delete anything (this will happen if you are running your update script the second time). This allows for better debugging of the scripts.

Advices (Cont. 1)

- ❑ Your update script may run quickly or take a long time, depending upon the intensity of updates in the system. You should design your scripts in such way that they will not run into each other.

- ❑ It is a given that you are going to make a lot of mistakes before setting everything up “just so”, so the following script can be used to undo the changes and start over (See etl_undo_cdc.sql).

Overview of CDC process



Create Change Tables

Create Subscription

Activate Subscription

Extend Subscription Window

Drop CDC Views

Purge CDC Window

Extend Subscription Window

Create CDC Views

Process CDC Views

Drop Subscription

Drop Change Tables

Questions and answers

Contact Information

Boris Knizhnik

BIK Information Services, Inc.

e-mail: borisk@bikinfo.com

Ph: 240-453-9510