

Getting Data Moving With Informix Enterprise Replication



XDB
SYSTEMS



Tom Beebe
tom@xdbsystems.com

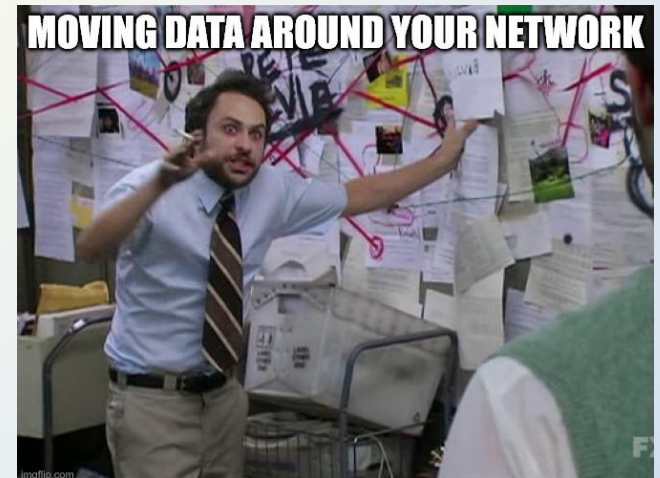
July 16th 2026

About This Webcast

- Intended as an introduction to ER
- Wanted a fairly simple step by step guide to getting up and running
- Explanation of the different Informix replication techniques and when you would want one versus the others

Agenda

- What is ER
- Differences between ER and HDR
- Setting up the Informix environment
- Defining the servers
- Defining simple replicates
- Multiple targets from a master server
- Checking the replication status
- Repairing replicates
- Questions



What Is Enterprise Replication

- Log based
- Asynchronous data replication
- Can be Primary-Target or update anywhere
- Efficient and Flexible
- Allows for data repair and synchronization
- Runs on multiple platforms

ER Vs HDR

HDR	Enterprise Replication
Complete instance replication	Replicates tables or even just rows
Does not replication BYTE/TEXT	Replicates BYTE/TEXT
(Default) synchronous replication	Asynchronous replication
Intended for DR	For HA, can be used for DR
All systems must be identical	Can be a mixed environment
Simple to set up	Can be complex to set up
Schema changes automatically replicated	Schema changes must be made individually
Network performance can impact primary	Replication status will not impact the primary server
No schema requirements	Requires PK, ERKeys or Unique Index

Reasons To Use ER

- Only want some of your data on the target server
- Setting up an update-anywhere environment
- Need replication but using BYTE or TEXT columns
- Mixed environment
- Unstable network between source and target servers
- Combining multiple targets into a single target server
- Feeding data from a source server into multiple instances
- Spreading your data and load across multiple instances

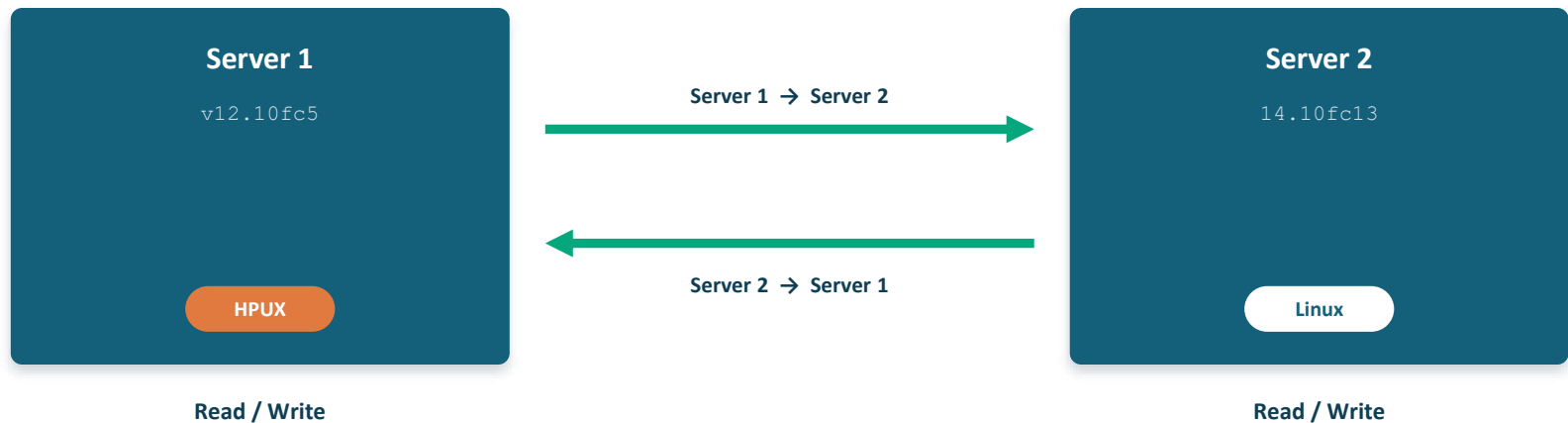
ER Replication Types

- Update anywhere
 - All systems are replicated r/w
 - An update at one point will (eventually) filter to all other nodes
- Primary-Target
 - All replicates are read only other than the master
 - Updates will filter down but not back up
 - Tables may not be in sync if there are changes made to target tables directly

Use Case: Update Anywhere

TOPOLOGY 1

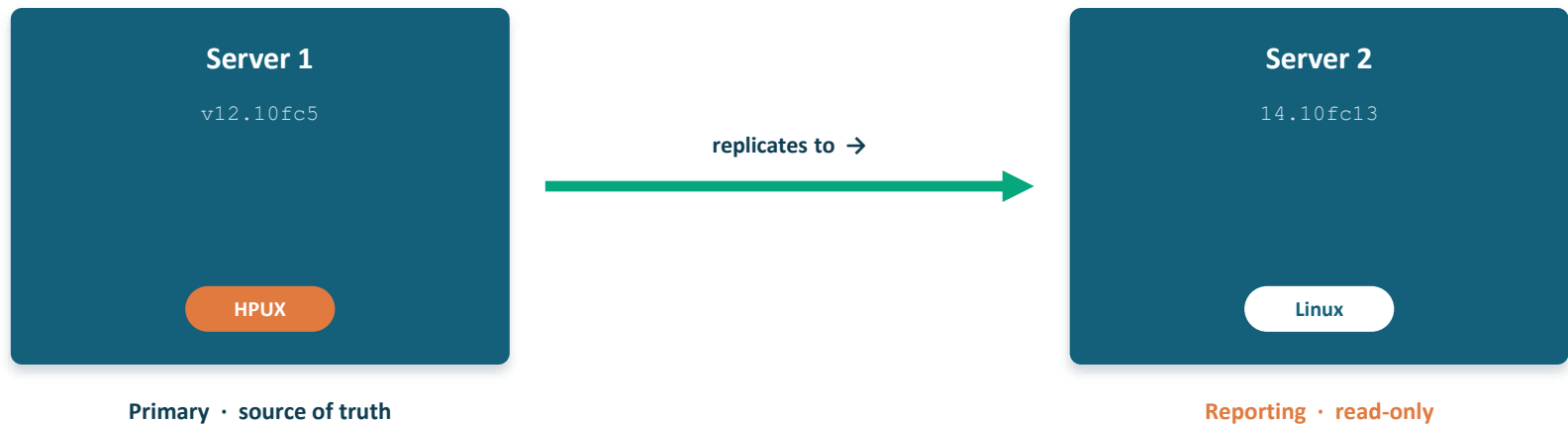
Bidirectional replication — writes on either server, keep the other up to date



Reporting Only Server

TOPOLOGY 2

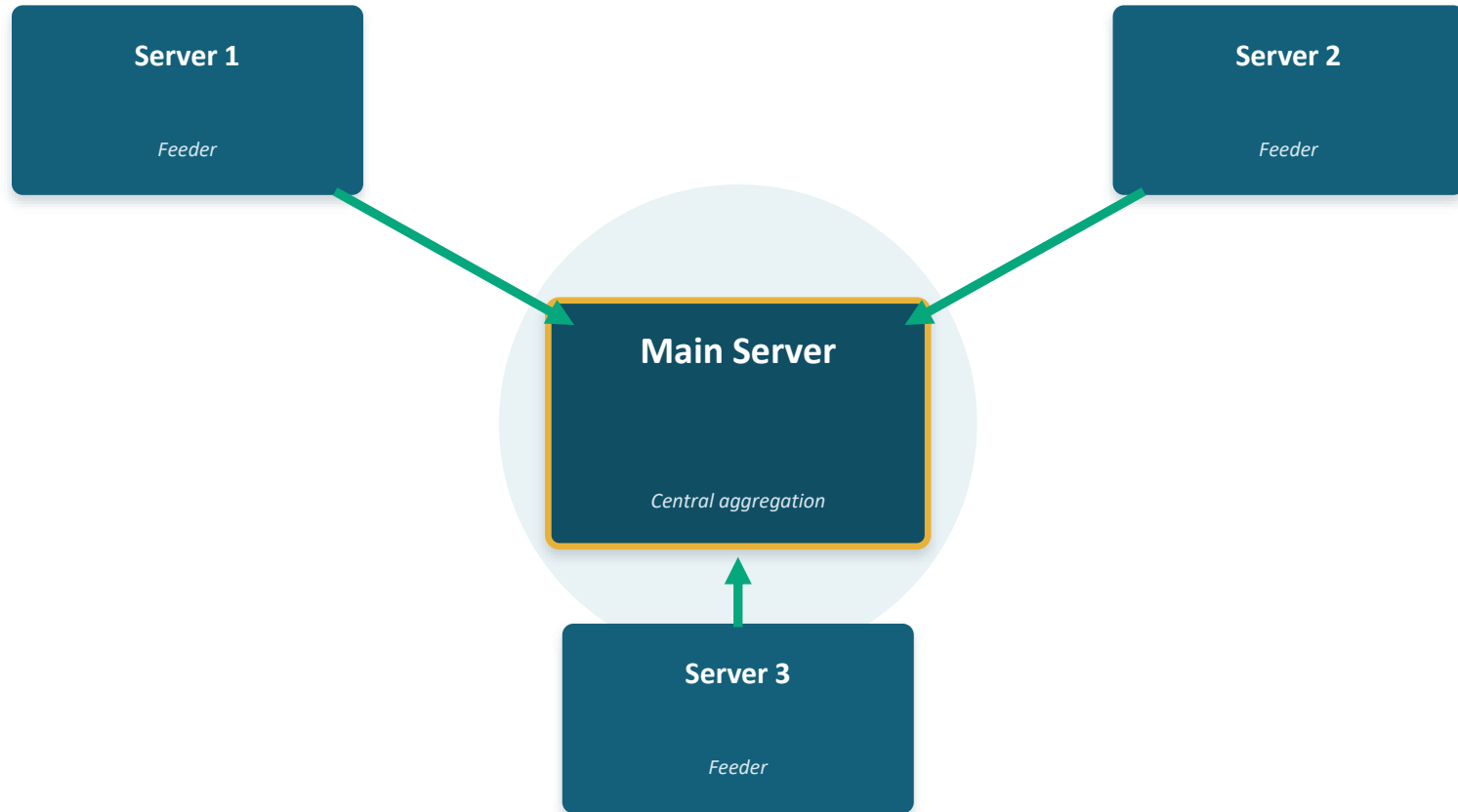
One-way replication — Server 1 feeds Server 2, to offload processing



Feeder Servers

TOPOLOGY 3

Hub-and-spoke — three feeder servers push data into the main server



Use Cases

Data Replication Across Servers

Showing different servers with different uses and what tables they need access to

TABLES ↓ / SERVERS →	Server #1 v12.10 HPUX Legacy Reporting System	Server #2 ★ v14.10fc12 Linux Central Warehouse	Server #3 v14.10fc3 Linux Feeder System Store 3	Server #4 v15 Linux Testing Server
stores on 3 of 4 servers	Readonly	RW	—	Readonly
sales on 3 of 4 servers	Readonly	Readonly	RW	—
customers on 2 of 4 servers	—	RW	Readonly	—
inventory on 3 of 4 servers	—	RW	RW	Readonly

★ **Replication hub:** Server #2 (Central Warehouse) is the only system holding all four tables — every other server shares a subset of its data.

ER Terminology

Domain

- An entire cluster of ER Nodes

Node

- A particular server

Replicate

- A table or set of columns that is being replicated

Replicate Set

- Group of replicates

ATS – Aborted Transaction Spooling

- Informix data on the full failed transaction

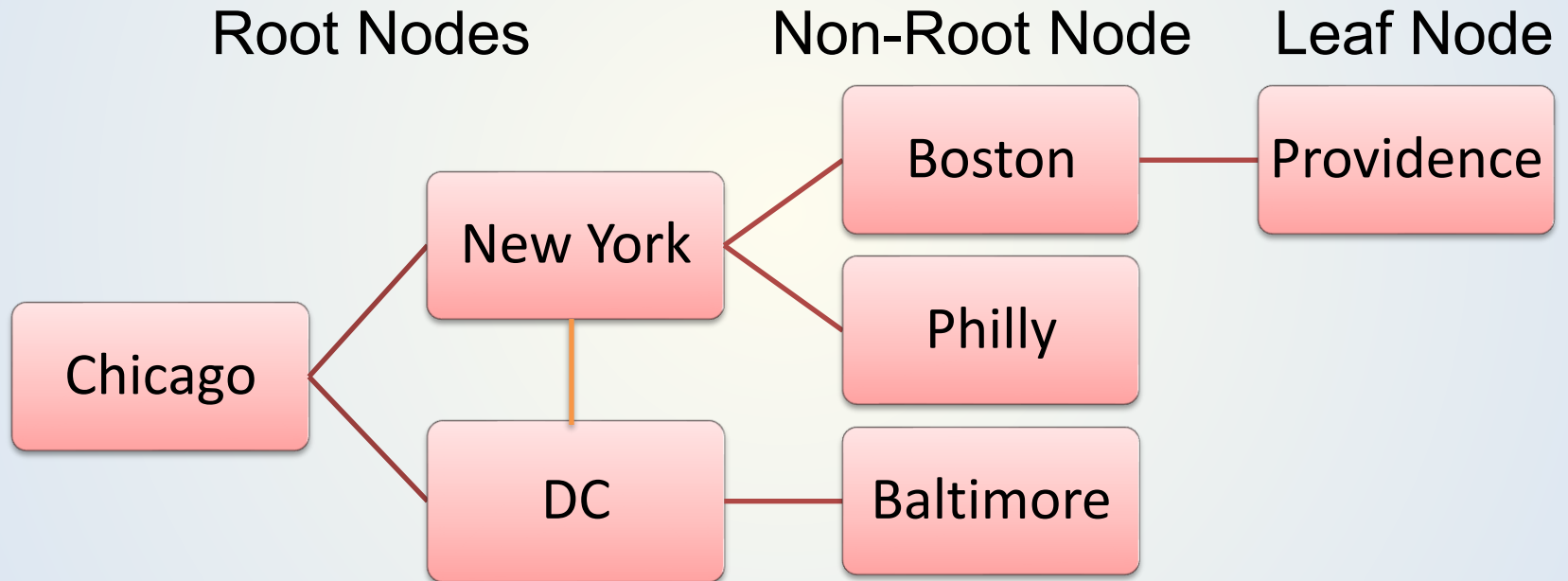
RIS – Row Information Spooling

- The row information for failed rows

ER Node Types

- Root Nodes
 - Connect to all attached servers
 - Will expect to replicate with all other root nodes
- Non-root Node
 - Will connect to root nodes and other leaf nodes
- Leaf Node
 - Only connects to local non-root or root node
 - Will not connect to other servers
 - Does not contain the full catalog of servers

Node Types



Conflict Resolution

- Options when defining the replicate:
 - Always: always apply the new record regardless
 - Ignore: Do not resolve conflicts
 - Timestamp: Row with the most recent update
 - Deletewins: Delete requests win, otherwise use timestamp
 - Stored Procedure: Use a SPL to determine which row takes precedence
 - Timestamp with SPL: Use a SPL only if the timestamps tie



Setup Steps For Informix

- Set up a sbspace to hold the send and receive queues
- Create a dbspace for transaction records
- Define a dbspace for syscdr database
- Verify connections between all servers that will replicate
- Verify there are sufficient logs on all of the systems
- Set up sqlhosts
- Set Informix to use tcp as DBSERVERNAME - important
- Make the ATS and RIS directories

Needed ONCONFIG Changes

- CDR_DBSPACE – where syscdr will be created, rootdbs is default
- CDR_QHDR_DBSPACE – where record headers will be stored
 - 110 bytes per record
 - Estimate records possibly created in 72 hours for sizing
- CDR_QDATA_SBSPACE – Where spooled transactions will be housed
 - Estimate at least 72 hours of transactions for storage
 - Can have more than one defined, comma separated

Trusted Connections

- Hosts.equiv, .rhosts,
- REMOTE_SERVER_CFG – version 12
 - Can add entries using sql
- Onpassword if it is an untrusted network

SQLHosts

- All ER connections need to be part of a group

```
er_grp      group  -      -      i=10
system_tcp  onsocket      systemname  port  g=er_grp
```

- Make sure DBSERVERNAME is the tcp connection and is the same as INFORMIXSERVER
- All servers must have each other listed in their sqlhosts files

Sample SQLHosts

g_chicago chicago_tcp	group onsoctcp	- chiserver	- sqlexec	i=1 g=g_chicago
g_ny ny_tcp	group onsoctcp	- nyserver	- sqlexec	i=2 g=g_ny
g_dc dc_tcp	group onsoctcp	- dcserver	- sqlexec	i=3 g=g_dc

Set Up Trusted Hosts

- Edit `~informix/.rhosts`
- Add `<hostname> Informix`
- Use `dbaccess` to verify that the servers can talk both ways
- If you get the error:
 - 951: Incorrect password or user `informix@dc.domain.com` is not known on the database server.
 - Add the listed hostname to the trusted hosts
 - The name resolution is from reverse dns from the IP

Setting Up The Data

- Make sure the table has a primary key
 - (v11 can use with ERKey)
 - (v12 can use –key to pass a unique index)
- Use ADD CRCOLS if using timestamp CR
- If you expect to use a good deal of data checking ADD REPLCHECK columns
- ALTER TABLE <table> ADD CRCOLS;
- ALTER TABLE <table> ADD ERKEY;

Define First Node

```
cdr define server \  
  -A $INFORMIXDIR/ats \  
  -R $INFORMIXDIR/ris \  
  -I g_chicago
```

- -A sets the ats directory
- -R sets the ris directory
- -I tells it to initialize the ER system
- The last parameter is the server being initialized

Define Second Node

```
cdr define server \  
  -A $INFORMIXDIR/ats \  
  -R $INFORMIXDIR/ris \  
  -I g_ny \  
  -S g_chicago
```

- -A sets the ats directory
- -R sets the ris directory
- -I tells it to initialize the ER system
- -S Tells it to synchronize the catalog with the defined server

CDR List Server

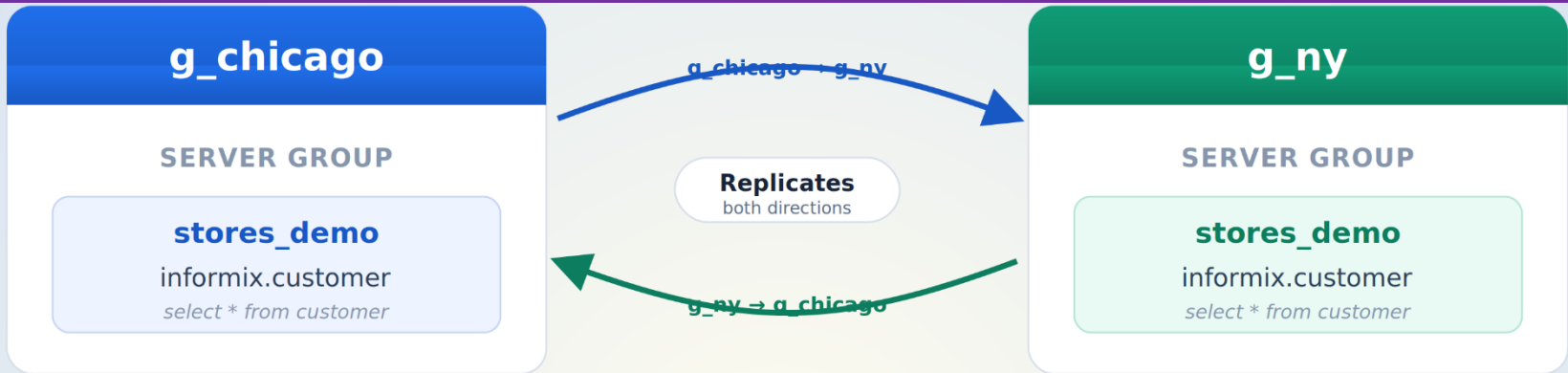
```
informix@chicago:~/scripts$ cdr list server
```

SERVER	ID	STATE	STATUS	QUEUE	CONNECTION
--------	----	-------	--------	-------	------------

CHANGED

g_chicago	1	Active	Local	0	
g_dc	3	Active	Connected	0	Dec 14 16:16:12
g_ny	2	Active	Connected	0	Dec 14 16:17:01

Define First Replicate



```
cdr define repl -c g_chicago -C always repl_customer \  
  "stores_demo@g_chicago:informix.customer" "select * from customer" \  
  "stores_demo@g_ny:informix.customer" "select * from customer"
```

- `cdr define repl` – tells cdr to define a replicate
- `-c` - tells it to connect to server `g_chicago` to run the command
- `-C` – this is the conflict resolution rule
- The first line is the first replicate source followed by the sql for it
- Second line is the second replicate followed by the sql
- Can have named columns instead of using `'*'`
- Can use different database or table names

Output

Interpreting this replicate as a master replicate.

Verification of stores_demo@g_chicago:informix.customer started

Verification of stores_demo@g_chicago:informix.customer is successful

Verification of stores_demo@g_ny:informix.customer started

Verification of stores_demo@g_ny:informix.customer is successful

cdr list repl

```
informix@chicago:~/scripts$ cdr list repl repl_customer
```

```
DEFINED REPLICATES ATTRIBUTES
```

```
-----
```

```
REPLICATE:          repl_customer
STATE:              Inactive ON:g_chicago
CONFLICT:           Always Apply
FREQUENCY:          immediate
QUEUE SIZE:         0
PARTICIPANT:        stores_demo:informix.customer
OPTIONS:            transaction,fullrow
REPLID:             65542 / 0x10006
REPLMODE:           PRIMARY  ON:g_chicago
APPLY-AS:           INFORMIX ON:g_chicago
REPLTYPE:           Master
```

Remote cdr list repl

```
informix@chicago:~/scripts$ cdr list repl -c g_ny repl_customer
```

DEFINED REPLICATES ATTRIBUTES

REPLICATE:	repl_customer
STATE:	Inactive ON:g_ny
CONFLICT:	Always Apply
FREQUENCY:	immediate
QUEUE SIZE:	0
PARTICIPANT:	stores_demo:informix.customer
OPTIONS:	transaction,fullrow
REPLID:	65542 / 0x10006
REPLMODE:	PRIMARY ON:g_ny
APPLY-AS:	INFORMIX ON:g_ny
REPLTYPE:	Master

Starting the Replicate

```
informix@chicago:~/scripts$ cdr start repl repl_customer
```

```
informix@chicago:~/scripts$ cdr list repl repl_customer
```

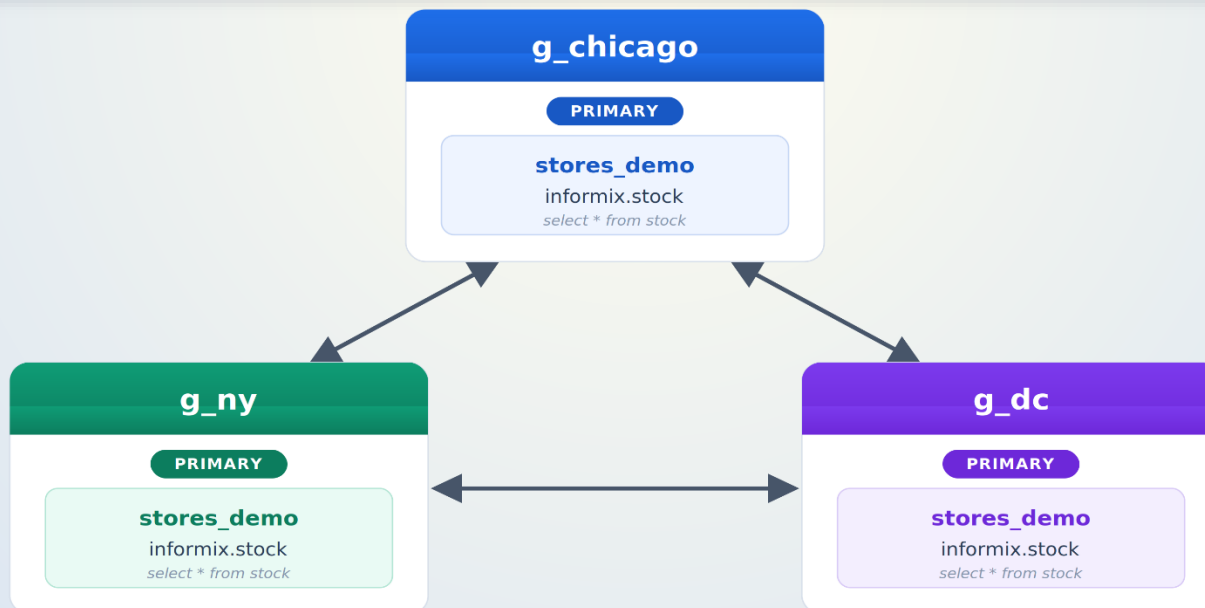
DEFINED REPLICATES ATTRIBUTES

```
REPLICATE:      repl_customer
STATE:          Active ON:g_chicago
CONFLICT:       Always Apply
FREQUENCY:      immediate
QUEUE SIZE:     0
PARTICIPANT:    stores_demo:informix.customer
OPTIONS:        transaction,fullrow
REPLID:         65542 / 0x10006
REPLMODE:       PRIMARY  ON:g_chicago
APPLY-AS:       INFORMIX ON:g_chicago
REPLTYPE:       Master
```

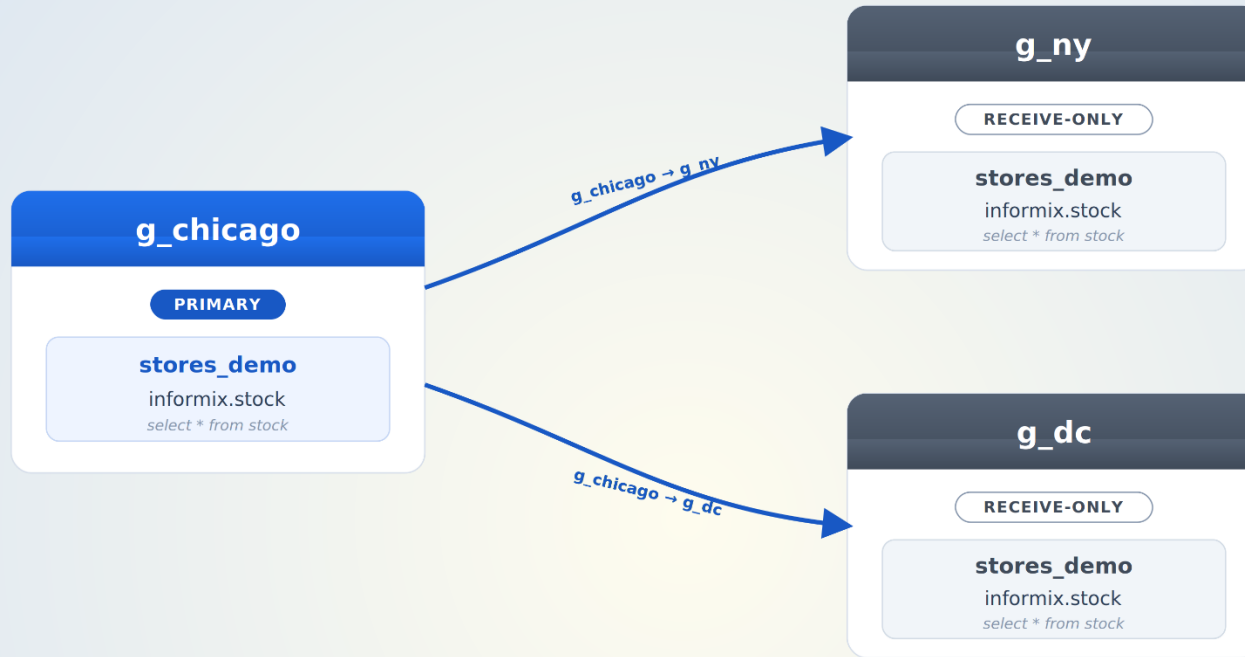
Multiple Targets Update Anywhere

```
cdr define repl -c g_chicago -C ignore repl_stock \  
"stores_demo@g_chicago:informix.stock" "select * from stock" \  
"stores_demo@g_ny:informix.stock" "select * from stock" \  
"stores_demo@g_dc:informix.stock" "select * from stock"
```

```
cdr start repl repl_stock
```



Primary-Target Example



```
cdr define repl -c g_chicago -C ignore repl_stock \  
  "P stores_demo@g_chicago:informix.stock" "select * from stock" \  
  "R stores_demo@g_ny:informix.stock" "select * from stock" \  
  "R stores_demo@g_dc:informix.stock" "select * from stock" \  
P = primary (source) R = receive-only (target)
```

```
cdr start repl repl_stock
```

Output from cdr list repl

```
informix@chicago:~/scripts$ cdr list repl -c g_dc
```

CURRENTLY DEFINED REPLICATES

```
-----  
REPLICATE:      repl_stock  
STATE:          Active ON:g_dc  
CONFLICT:       Ignore  
FREQUENCY:      immediate  
QUEUE SIZE:     0  
PARTICIPANT:    stores_demo:informix.stock  
OPTIONS:        transaction,fullrow  
REPLID:         65541 / 0x10005  
REPLMODE:       PRIMARY  ON:g_dc  
APPLY-AS:       INFORMIX ON:g_dc  
REPLTYPE:       Master
```

Sendq

```
informix@chicago:~/scripts$ cdr view sendq
```

```
RQM SENDQ
```

Server	Trans. in que	Trans. in mem	Trans. spooled	Data in queue	Memory in use	ACKS pending
g_chicago	1	1	0	113	113	0
g_dc	0	0	0	0	0	0
g_ny	0	0	0	0	0	0

Chicago - Primary

```
informix@chicago:~/scripts$ cdr list repl repl_stock
```

```
DEFINED REPLICATES ATTRIBUTES
```

```
-----
```

```
REPLICATE:          repl_stock
STATE:              Active ON:g_chicago
CONFLICT:           Ignore
FREQUENCY:          immediate
QUEUE SIZE:         0
PARTICIPANT:        stores_demo:informix.stock
OPTIONS:            transaction,fullrow
REPLID:             65543 / 0x10007
REPLMODE:           PRIMARY  ON:g_chicago
APPLY-AS:           INFORMIX ON:g_chicago
REPLTYPE:           Master
```

NY - Target

```
informix@chicago:~/scripts$ cdr list repl -c g_ny repl_stock
```

```
DEFINED REPLICATES ATTRIBUTES
```

```
-----
```

```
REPLICATE:      repl_stock
STATE:          Active ON:g_ny
CONFLICT:       Ignore
FREQUENCY:      immediate
QUEUE SIZE:     0
PARTICIPANT:    stores_demo:informix.stock
OPTIONS:        transaction,fullrow
REPLID:         65543 / 0x10007
REPLMODE:       READ-ONLY ON:g_ny
APPLY-AS:       INFORMIX ON:g_ny
REPLTYPE:       Master
```

Replicate Status Check

```
cdr check replicate --master=g_chicago --repl=repl_stock g_dc g_ny
```

- --master is the primary server to check from
- --repl is the replicate name
- The trailing options are any servers to check
- Large tables can take a while

```
Dec 14 2017 16:57:31 ----- Table scan for repl_stock start -----
```

Node	Rows	Extra	Missing	Mismatch	Processed
g_chicago	74	0	0	0	0
g_dc	74	0	0	0	0
g_ny	74	0	0	0	0

Repairing Replicates

```
cdr check replicate --master=g_chicago \  
--repl=repl_stock --repair --all
```

- Similar syntax but adding -repair fixes mistakes
- --all will check all replicates
- Treats the master as the primary system, it will remove and update rows on all of the other replicates to match the master, so make sure to use the correct master
- Will use logical logs

cdr check flags

- --background – Runs the command in the background
- --name – Sets a name for the check job, needed to look up the status
- --extratargetrows – allows you to define how to handle extra rows found on the target
- --firetrigger – How to handle triggers on the target systems
- --skipLOB – Do not compare large object data
- --timestamp – Instead of master you can base the updates on what row has the most recent timestamp
- --where – Use a where clause to only look at specific parts of the data

cdr sync replicate

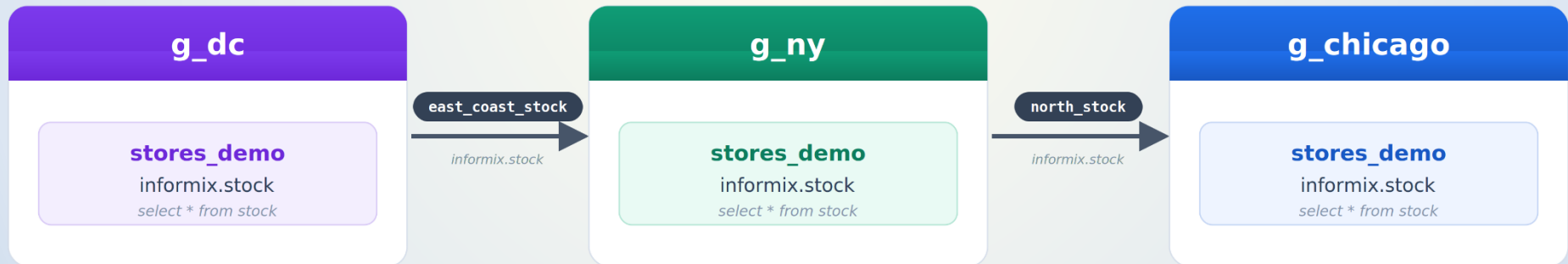
- Used to copy missing data to target replicates
- Meant to add missing rows
- Bypasses logical logs
- Good to use if populating a table for the first time

Other Notes

- Most ER commands can be run through task/admin commands in sysadmin
- ER can be set up using SSL connections
- Extensive monitoring commands (onstat -g cdr, onstat -g rqm, etc)
- cdr error will show any replication errors
- Templates of replicates can be created to easily add new systems to the network

Cascading Data

- By default, when there are chained replicates data changes will only go to the first node
- With cascading enabled, it will replicate through the chain



More Notes

- Replicatesets can be created for groups of replicates which allows for managing of those replicates with a single command
- Replicates and ER can be both paused and stopped and started as needed
- ER Grid was introduced to allow for easier management of systems
- If running versions prior to 11 note that while it can replicate with version 11+ systems there can be unexpected issues
- Can set ER up to only replicate on a schedule

Informix Version Notes

- 7.22 – ER Introduced
- 9.3 – UDT and parallel apply added
- 9.40 – Large transaction, HDR and Encryption support
- 10 – Templates, sync/check
- 11.10 – Truncate, onconfig change support, extensive bug fixes
- 11.70 – Flexible grid, cloning ER with ifxclone, erkey
- 12.10 – Timeseries, sharding support, removed primary key requirement
- 14 – Self replication, replication to SPL, large performance improvements
- 15 – Introduced cascading replication for templates

Links

- Setting up Informix ER For the First Time
 - <http://www-01.ibm.com/support/docview.wss?uid=swg21153338>
- Creating a diagram for Informix ER
 - <https://www.ibm.com/developerworks/data/tutorials/dm-1204informixerdiagram/index.html>
- Nagaraju Inturi's ER Presentation To The WAIUG
 - http://waiug.org/wp-content/uploads/2017/08/Nagaraju_Inturi_ER_WAIUG_Aug_2017.pdf
- ONCONFIG configuration parameters
 - https://www.ibm.com/support/knowledgecenter/en/SSGU8G_12.1.0/com.ibm.adref.doc/ids_adr_1072.htm#ids_adr_1072_er
- CDR Cascade Information
 - https://www.ibm.com/docs/en/informix-servers/15.0.x?topic=utility-cdr-define-replicate#ids_erp_300__spl_opt_cascaderepl

Questions?

**Send follow-up questions
to tom@xdbsystems.com**

Certifications



Thank You

Tom Beebe

tom@xdbsystems.com

For more information:

<https://www.xdbsystems.com>

XDB SYSTEMS

The IBM logo, consisting of the letters "IBM" in a bold, sans-serif font, with a registered trademark symbol (®) to the right.

Silver Partner

Seeking TechTalk Speakers



- We are looking for additional speakers and especially any new speakers to share your Informix knowledge with our user Community via the IIUG Tech Talks. We will assist you in preparing for your Tech Talks presentation by prerecording your session. This will allow you to be available to answer any questions that come up in the chat during you presentation.
- If you are interested in producing a 15 – 45-minute presentation, please contact either Tom Beebe or Bruce Simms for additional information. (tom@iiug.org or bruce@iiug.org)

IIUG Tech Talk:

TimeSeries implementation using
IBM Bob with Jan Musil

Aug 20 2026 1pm EDT

Migration off of Informix 4GL
“Green Screen” - with Jan Musil

Sept 17th 2026 1pm EDT



International
Informix
Users Group

