SYNERGY DEVPARTNER CONFERENCE 2026

ISAM Rev 7

Familiar records. A new foundation.

Jeff Greene, CTO

Keep the knowledge
in your DBL application.

Give its data more protection,
visibility and possibilities.

SYNERGY/DE • ISAM REV 7 BETA



Keep established applications moving.

Two investments in the software your business already depends on.

REV 6 / HIGH-COOPERATIVE FILES

Extend records. Change keys. Keep the application running.



HICO changes a live file's structure. Applications continue reading and writing.

HICO · Wed 9:00 AM · Tate Chamberlain

REV 7 / NATIVE DATABASE TABLES

Keep record access. Expand the storage foundation.



Shared policy, recovery and operational signals.

The mapping turns around.

The same engine connects SQL and records. The stored side changes.

xfODBC REPLACEMENT

SQL over records

SQL ROW REPRESENTATION

42	WEST	12.50

mapping

0042	WEST	001250

Stored in Rev 6 .ISM / .IS1 files
DBL continues to use those files.

ISAM REV 7

Records over rows

DBL RECORD REPRESENTATION

0042	WEST	001250

mapping

42	WEST	12.50

Stored in native GantrySQL tables
SQL and DBL access those rows.

A familiar record. A native database row.

An explicit mapping preserves the layout your DBL program expects.

DBL PROGRAM

READ / READS
STORE / WRITE

0042	WEST	001250
0–3 · d4	4–7 · a4	8–13 · d6.2

RECORD LAYOUT ↔ TYPED COLUMNS

Encode on READ.
Decode on WRITE.

STORED HERE

id	region	balance
42	WEST	12.50

NATIVE GANTRYSQL TABLE

Repository
↓ MetaDataTool
↓ Validated map

```
SELECT id, region,
       balance
FROM customer
WHERE id = 42;
```

SQL reads and updates the same native row.

One native URI. Three deployment configurations.

Native access with a GantrySQL-enabled Synergy 12.6+ runtime

`gantry://db/customer`

OPEN · READ · WRITE

Same machine



Shared memory

Selected at OPEN

RDMA configured



RDMA

Selected at OPEN

RDMA unavailable



QUIC

Selected at OPEN

Separate route: Synergy 10+ · customer@dbhost · xfServer protocol

Your DBL lock coordinates SQL writers, too.

Two applications. The same customer record.

DBL PROGRAM

Channel open for update

READ customer 42

Add a $0.50 fee

WRITE

CUSTOMER 42

Balance **13.00**

WRITE published

Lock released

SQL APPLICATION

```
UPDATE customer
SET region = 'EAST'
WHERE id = 42;
```

- **Writer can proceed**

WRITE publishes the change and releases the automatic lock.

PLANNED Explicit DBL transactions spanning multiple records.

Same row. Different views of change.

Customer 42 · two separate examples, starting at 12.50

DBL publishes a change → SQL readers keep their own view

Existing SQL snapshot REPEATABLE READ	→	DBL WRITE New value published	→	New SQL statement READ COMMITTED
12.50 Established first		13.00		13.00

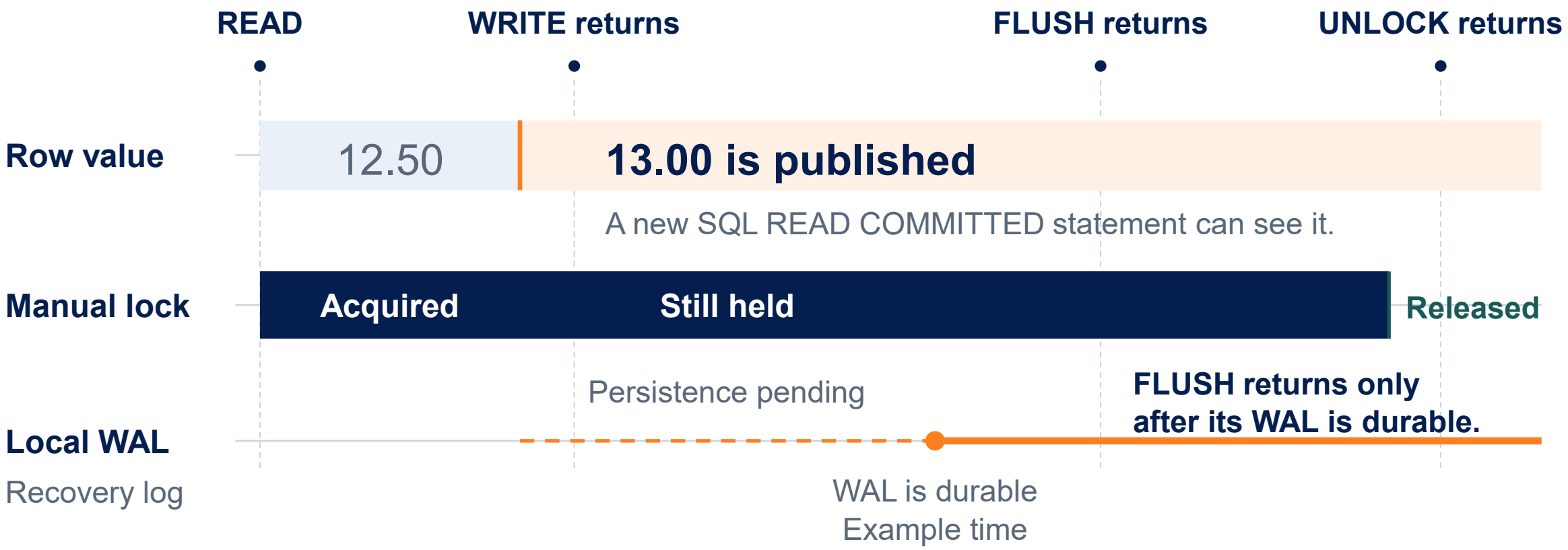
SQL changes a value → DBL reads WITHOUT a lock

SQL changes balance Not committed	→	Unlocked DBL READ Value received	→	SQL ROLLBACK Stored balance restored
13.00		13.00 Already observed		12.50

For a settled handoff: SQL COMMIT, then signal the record work.

Visible. Locked. Durable.

Explicit manual lock · default durability batching



Batching off: each mutation waits for local WAL before returning.

The Synergy SELECT class gets faster.

Directional findings from Synergex testing

Synergy SELECT class

Generally much faster

Across transports

CURRENT BASELINE: xfServer

Rev 7 generally meets or exceeds

CURRENT BASELINE: local ISAM

Reads	Generally faster
Updates / writes	Slightly slower

Compare your workload at the same durability boundary.

Control the rows. Protect their stored bytes.

Who can use these rows?

West-branch role

DBL READ

SQL SELECT

Native customer table

42	WEST	12.50
43	EAST	8.00
44	NORTH	19.00

SELECT allowed · writes denied · region = WEST

Both doors return the permitted rows.

Table privileges + enabled row policies

Who can read the stored bytes?

Native data + WAL

Encryption set at creation

Encrypted backups

Separate backup key configuration

Keys under your control

Local file · AWS KMS · Vault

Recover saved state. Restore a serving leader.

Process stops. Storage survives.

Illustrative page and durable log

Data page	Durable WAL
12.50	→ 13.00

The page has not caught up.
The log has the durable change.

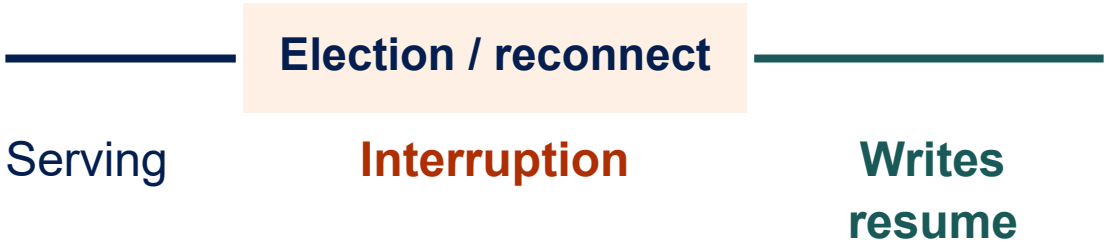
Recovered row: 13.00

No durable commit → unfinished work undone.

One node lost. Majority survives.



Two surviving voters can elect a leader.



Writes require a majority and client recovery.

Rebuild the rows. Hand over the live identity.

Example · native orders in one replication group · recovery target 09:40

Application traffic

DBL + SQL

Reopen / resume

Full backup

Retained log through 09:40

Two physical tables

Prior orders copy · retained

0 orders

Retained for review

orders **LIVE**

```
1042   OPEN     $125.00
1043   READY     $80.00
```

Other tables stay current. Validate related data.

A controlled handoff

1 Rebuild separately

Earlier rows + indexes.
Live table stays in place.

2 Coordinate writers

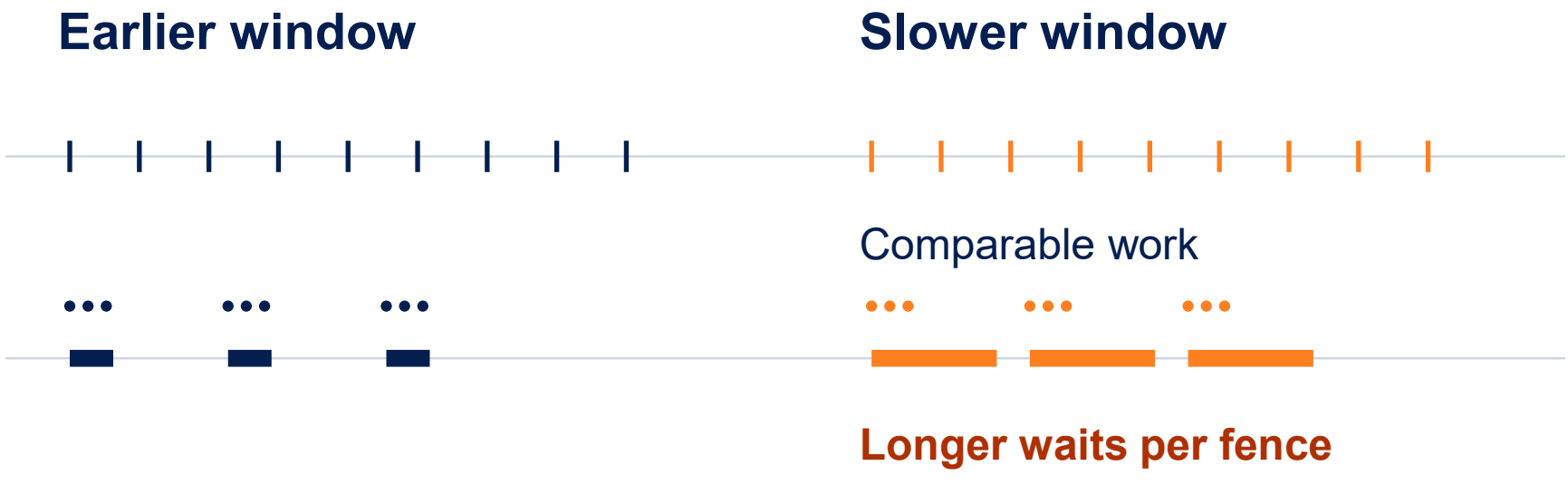
Finish active writers.
Hold new writers for the swap.

3 Resume on rebuilt data

Reopen affected DBL channels, then resume.

The posting run slows. Where is the time going?

Illustrative posting workload · equal observation windows

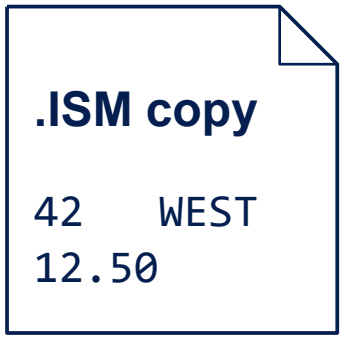


Start with WAL / storage pressure. Correlate with load and failures.

Configure an alert for rising fence waits.

Start with one file. Prove the workflow.

1 Copy one file



Use a file copy with known values.

2 Match the values

Record bytes

`0042WEST001250`

`42 WEST 12.50`

Compare field values and key behavior.

3 Exercise the workflow

DBL **SQL**

Same native row

Test results, contention and durability fences.

4 Verify the restore

Restored customer

`42 WEST 12.50`

Backup → restore

Reopen in the application. Compare recovered values.

No additional Rev 7 license for supported Synergy/DE customers.

Keep the knowledge. Build what comes next.

YOUR EXISTING WORKFLOW

Post a payment

DBL · READ → WRITE

A SERVICE YOU COULD BUILD

Branch enquiry

SQL · West-branch role

NATIVE CUSTOMER TABLE

```
42    WEST    12.50
43    EAST     8.00
```

Row policy · Durable history

```
42   WEST   12.50
```

West customers only

Build the next thing your customers need.

Start with the application they already trust.

Your application knowledge **Our database engineers**

Choose the first workload with us.

Thank you for 50 years of building with us.