

diffst - A tool for Data Integration Testing

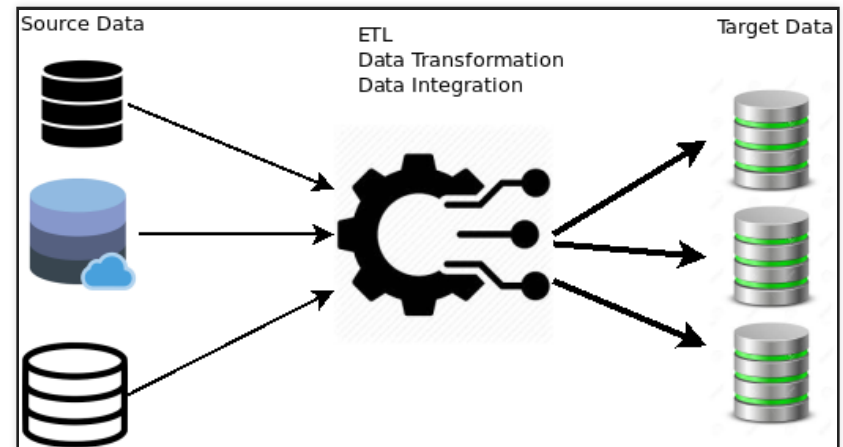
Francois Marais
francois@busii.com

Outline

- Data Integration Applications
- Common risks and challenges
- Functional testing approaches
- diffst - Features and limitations
- Questions and Discussion

Data Integration/Migration Applications

- Once-off projects
 - Data Migration - hardware, software, database upgrade/relocation
 - Application Migration (CRM/ERP)
 - Take-on of new data sets (eg corporate mergers/acquisitions)
- Systems and Applications
 - Data Warehousing
 - Business Intelligence, Big Data,...
 - Loyalty Programme Engines



Data Integration Applications - Typical Risks

- Data not extracted from Source as expected
 - Wrong time period
 - Not all expected data extracted
 - Extracted at wrong time
- Expected business rules not applied
 - Fields truncated
 - Data Types misread
- Target data Incomplete/Duplicated/Inconsistent..
- Wrong grain

Coverage Challenges

- Large actual data volumes and
- Business logic complexity - per column
- Implies many actual data points available for testing
- ... and they need to be tested - more important than synthetic test data

Common functional testing checks

- SELECT data from Source
- Apply business rules
- SELECT expected data from Target
- Compare and evaluate
- \Rightarrow Automation opportunity

Sampling: Comparing individual data items

The image shows two side-by-side database query windows. The left window is titled '*<MySQL 8+ - tpcds> Script-1' and contains a complex SQL query. The right window is titled '*<SQLite - x.db> Script' and contains a simpler SQL query. Both windows show the results of their respective queries in a grid view.

Left Window (MySQL 8+ - tpcds):

```
select s_name,  
       n_name,  
       case when p_partkey=197456 then 234 else p_partkey end,  
       p_mfgr  
from PART,  
     SUPPLIER,  
     PARTSUPP,  
     NATION,  
     REGION  
where p_partkey = ps_partkey  
and s_suppkey = ps_suppkey  
and s_nationkey = n_nationkey  
and n_regionkey = r_regionkey  
and s_name='Supplier#000009976'  
and p_partkey <> 9975  
order by s_name,n_name,p_partkey,p_mfgr
```

Right Window (SQLite - x.db):

```
SELECT acctbal, name, country, partkey, mfgr  
FROM supplier_detail;
```

Results:

Left Window Results (Supplier#000009976):

Grid	asc s_name	asc n_name	123 case when p_partkey=197456 then 234 else p_partkey end	asc p_mfgr
1	Supplier#000009976	ALGERIA	2 475	Manufacturer#1
2	Supplier#000009976	ALGERIA	4 975	Manufacturer#2
3	Supplier#000009976	ALGERIA	7 475	Manufacturer#4
4	Supplier#000009976	ALGERIA	12 472	Manufacturer#5
5	Supplier#000009976	ALGERIA	14 973	Manufacturer#4

Right Window Results (supplier_detail):

Grid	123 acctbal	asc name	asc country	123 partkey	asc mfgr
1	3,0600000000	Supplier#000009976	ALGERIA	2 475	Manufacturer#1
2	3,0600000000	Supplier#000009976	ALGERIA	4 975	Manufacturer#2
3	3,0600000000	Supplier#000009976	ALGERIA	7 475	Manufacturer#4
4	3,0600000000	Supplier#000009976	ALGERIA	9 975	Manufacturer#5
5	3,0600000000	Supplier#000009976	ALGERIA	12 472	Manufacturer#5
6	3,0600000000	Supplier#000009976	ALGERIA	14 973	Manufacturer#4
7	3,0600000000	Supplier#000009976	ALGERIA	17 474	Manufacturer#4
8	3,0600000000	Supplier#000009976	ALGERIA	19 975	Manufacturer#4
9	3,0600000000	Supplier#000009976	ALGERIA	22 469	Manufacturer#2
10	3,0600000000	Supplier#000009976	ALGERIA	24 971	Manufacturer#2
11	3,0600000000	Supplier#000009976	ALGERIA	27 473	Manufacturer#1
12	3,0600000000	Supplier#000009976	ALGERIA	29 975	Manufacturer#5
13	3,0600000000	Supplier#000009976	ALGERIA	32 466	Manufacturer#2

- Useful for testing specific data items - Exploratory value
- But not effective in testing large data volumes
- How to build into regression, test suites?

Aggregated SQL

The image shows two side-by-side screenshots of SQL IDEs. The left IDE is MySQL 8+ and the right is SQLite. Both show a SQL query for summing account balances and a result grid. In the MySQL result grid, the value 241,74 is circled in red. In the SQLite result grid, the value 244,8 is circled in red. This visual comparison highlights a discrepancy between the two databases' aggregation results.

MySQL 8+ - tpcds> Script-1

```
select sum(s_acctbal)
  from PART,
       SUPPLIER,
       PARTSUPP,
       NATION,
       REGION
 where p_partkey = ps_partkey
       and s_suppkey = ps_suppkey
       and s_nationkey = n_nationkey
       and n_regionkey = r_regionkey
       and s_name='Supplier#000009976'
       and p_partkey <> 9975
```

Result

	123 sum(s_acctbal)
1	241,74

SQLite - x.db> Script

```
select sum(acctbal) from supplier_detail
```

Result

	123 sum(acctbal)
1	244,8

- Can only test few columns at a time
- Slow to investigate source-target discrepancies

MINUS Queries

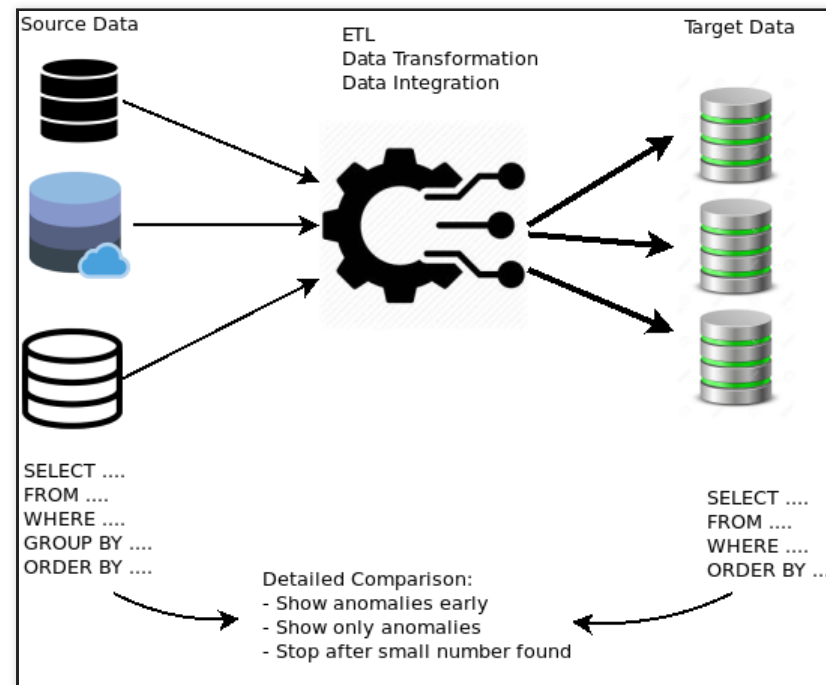
- Source and Target need to be accessed in same statement
- Slow - have to run a query on each side separately - then compare

Source				Target			
	Col1	Col2	Col3		Col1	Col2	Col3
Row1	xxxxx	xxxxx	xxxxx	Row1	xxxxx	xxxxx	xxxxx
Row2	xxxxx	xxxxx	xxxxx	Row2	xxxxx	xxxxx	xxxxx
Row3	xxxxx	xxxxx	xxxxx	Row3	xxxxx	xxxxx	xxxxx
Row4	xxxxx	xxxxx	xxxxx	Row4	xxxxx	xxxxx	xxxxx
Row5	xxxxx	xxxxx	xxxxx	Row5	xxxxx	xxxxx	xxxxx
Row6	xxxxx	xxxxx	xxxxx	Row6	xxxxx	xxxxx	xxxxx
Row7	xxxxx	xxxxx	xxxxx	Row7	xxxxx	xxxxx	xxxxx
Row8	xxxxx	xxxxx	xxxxx	Row8	xxxxx	xxxxx	xxxxx
Row9	xxxxx	xxxxx	xxxxx	Row9	xxxxx	xxxxx	xxxxx
Row10	xxxxx	xxxxx	xxxxx	Row10	xxxxx	xxxxx	xxxxx
Row11	xxxxx	xxxxx	xxxxx	Row11	xxxxx	xxxxx	xxxxx
Row12	xxxxx	xxxxx	xxxxx	Row12	xxxxx	xxxxx	xxxxx
Row13	xxxxx	xxxxx	xxxxx	Row13	xxxxx	xxxxx	xxxxx
....	xxxxx	xxxxx	xxxxx	xxxxx	xxxxx	xxxxx

Manual Detailed Comparison							
Source				Target			
	Col1	Col2	Col3		Col1	Col2	Col3
Row1	xxxxx	xxxxx	xxxxx	Row1	xxxxx	xxxxx	xxxxx
Row2	xxxxx	xxxxx	xxxxx	Row2	xxxxx	xxxxx	xxxxx
Row3	xxxxx	xxxxx	xxxxx	Row3	xxxxx	xxxxx	xxxxx
Row4	xxxxx	xxxxx	xxxxx	Row4	xxxxx	xxxxx	xxxxx
Row5	xxxxx	xxxxx	xxxxx	Row5	xxxxx	xxxxx	xxxxx
Row6	xxxxx	xxxxx	xxxxx	Row6	xxxxx	xxxxx	xxxxx
Row7	xxxxx	xxxxx	xxxxx	Row7	xxxxx	xxxxx	xxxxx
Row8	xxxxx	xxxxx	xxxxx	Row8	xxxxx	xxxxx	xxxxx
Row9	xxxxx	xxxxx	xxxxx	Row9	xxxxx	xxxxx	xxxxx
Row10	xxxxx	xxxxx	xxxxx	Row10	xxxxx	xxxxx	xxxxx
Row11	xxxxx	xxxxx	xxxxx	Row11	xxxxx	xxxxx	xxxxx
Row12	xxxxx	xxxxx	xxxxx	Row12	xxxxx	xxxxx	xxxxx
Row13	xxxxx	xxxxx	xxxxx	Row13	xxxxx	xxxxx	xxxxx
....	xxxxx	xxxxx	xxxxx	xxxxx	xxxxx	xxxxx

Group By Query Comparison							
Source				Target			
	Col1	Col2	Col3		Col1	Col2	Col3
Row1	xxxxx	xxxxx	xxxxx	Row1	xxxxx	xxxxx	xxxxx
Row2	xxxxx	xxxxx	xxxxx	Row2	xxxxx	xxxxx	xxxxx
Row3	xxxxx	xxxxx	xxxxx	Row3	xxxxx	xxxxx	xxxxx
Row4	xxxxx	xxxxx	xxxxx	Row4	xxxxx	xxxxx	xxxxx
Row5	xxxxx	xxxxx	xxxxx	Row5	xxxxx	xxxxx	xxxxx
Row6	xxxxx	xxxxx	xxxxx	Row6	xxxxx	xxxxx	xxxxx
Row7	xxxxx	xxxxx	xxxxx	Row7	xxxxx	xxxxx	xxxxx
Row8	xxxxx	xxxxx	xxxxx	Row8	xxxxx	xxxxx	xxxxx
Row9	xxxxx	xxxxx	xxxxx	Row9	xxxxx	xxxxx	xxxxx
Row10	xxxxx	xxxxx	xxxxx	Row10	xxxxx	xxxxx	xxxxx
Row11	xxxxx	xxxxx	xxxxx	Row11	xxxxx	xxxxx	xxxxx
Row12	xxxxx	xxxxx	xxxxx	Row12	xxxxx	xxxxx	xxxxx
Row13	xxxxx	xxxxx	xxxxx	Row13	xxxxx	xxxxx	xxxxx
....	xxxxx	xxxxx	xxxxx	xxxxx	xxxxx	xxxxx

Test Tool – Detailed Comparison of entire data sets								
Source				Target				
	Col1	Col2	Col3		Col1	Col2	Col3	
Row1	xxxxx	xxxxx	xxxxx	Row1	xxxxx	xxxxx	xxxxx	
Row2	xxxxx	xxxxx	xxxxx	Row2	xxxxx	xxxxx	xxxxx	
Row3	xxxxx	xxxxx	xxxxx	Row3	xxxxx	xxxxx	xxxxx	
Row4	xxxxx	xxxxx	xxxxx	Row4	xxxxx	xxxxx	xxxxx	
Row5	xxxxx	xxxxx	xxxxx	Row5	xxxxx	xxxxx	xxxxx	
Row6	xxxxx	xxxxx	xxxxx	Row6	xxxxx	xxxxx	xxxxx	
Row7	xxxxx	xxxxx	xxxxx	Row7	xxxxx	xxxxx	xxxxx	
Row8	xxxxx	xxxxx	xxxxx	Row8	xxxxx	xxxxx	xxxxx	
Row9	xxxxx	xxxxx	xxxxx	Row9	xxxxx	xxxxx	xxxxx	
Row10	xxxxx	xxxxx	xxxxx	Row10	xxxxx	xxxxx	xxxxx	
Row11	xxxxx	xxxxx	xxxxx	Row11	xxxxx	xxxxx	xxxxx	
Row12	xxxxx	xxxxx	xxxxx	Row12	xxxxx	xxxxx	xxxxx	
Row13	xxxxx	xxxxx	xxxxx	Row13	xxxxx	xxxxx	xxxxx	
....	xxxxx	xxxxx	xxxxx	xxxxx	xxxxx	xxxxx	



Requirement for Test Tool

An In-House Solution - Based on Unix Philosophy

- Write programs that do one thing and do it well.
- Write programs to work together.
- Write programs to handle text streams, because that is a universal interface.

Peter H. Salus, A Quarter-Century of Unix (1994)

diffst - Test Case

```
1 Source:
2   DB: SUPPDB
3   SQL: |
4       select s_name,
5              n_name,
6              p_partkey,
7              max(p_mfgr) mfgr
8   from PART,
9        SUPPLIER,
10       PARTSUPP,
11       NATION,
12       REGION
13  where p_partkey = ps_partkey
14         and s_suppkey = ps_suppkey
15         and s_nationkey = n_nationkey
16         and n_regionkey = r_regionkey
17         and s_name='Supplier#000009976'
18         -- and p_partkey=[< SINGLE >]
19  group by s_name,n_name,p_partkey
20  order by s_name,n_name,p_partkey
21 Target:
22   DB: DW
23   SQL: |
24       select name, country,
25              partkey,mfgr
26  from supplier_detail
27     -- where partkey=[< SINGLE >]
28  order by name, country, partkey,mfgr
29 Description: Supplier details by manufacturer and country
```


Config in the Environment

```
# The Data Warehouse DB Connection
export DIFFST_DB_DW_DRIVER=sqlite3
export DIFFST_DB_DW_URL=/home/fm/x.db

export DIFFST_DB_ACS_URL=/home/fm/x.db
export DIFFST_DB_ACS_DRIVER=sqlite3
export DIFFST_DB_MYSQLEG_DRIVER=mysql
export DIFFST_DB_MYSQLEG_URL=fm:lkj@/mysql

# Supplier Details Database
export DIFFST_DB_SUPPDB_DRIVER=mysql
export DIFFST_DB_SUPPDB_URL='tpcds:tpcds@tcp(127.0.0.1:3306)/tpcds'

# Test Case Directory
export DIFFST_TC_DIR=/home/fm/t
```

diffst - Output

```
fm@fm-VirtualBox:~$ diffst -t supplier_details
Setting up SUPPDB DB connection:...
Running SUPPDB SQL...
Setting up DW DB connection:...
Running DW SQL...
Fetching results...
+-----+-----+
| Supplier#000009976|ALGERIA|62457|Manufacturer#5 | >Supplier#000009976|ALGERIA|9975|Manufacturer#5
| Supplier#000009976|ALGERIA|184939|Manufacturer#5 | <
| Supplier#000009976|ALGERIA|197456|Manufacturer#9 | *Supplier#000009976|ALGERIA|184939|Manufactur
| Supplier#000009976|ALGERIA|197456|Manufacturer#9 | *Supplier#000009976|ALGERIA|197456|Manufacturer#4
+-----+-----+
2019/08/14 23:38:54 Not OK - diffs found
```

diffst - Output

```

fm@fm-VirtualBox:~$ diffst -t supplier_details -c
Setting up SUPPDB DB connection:...
Running SUPPDB SQL...
Setting up DW DB connection:...
Running DW SQL...
Fetching results...

```

name		>Supplier#000009976
country		>ALGERIA
partkey		>9975
mfgr		>Manufacturer#5
name	Supplier#000009976	<
country	ALGERIA	<
partkey	62457	<
mfgr	Manufacturer#5	<
name	Supplier#000009976	Supplier#000009976
country	ALGERIA	ALGERIA
partkey	184939	184939
mfgr	Manufacturer#5	*Manufactur
name	Supplier#000009976	Supplier#000009976
country	ALGERIA	ALGERIA
partkey	197456	197456
mfgr	Manufacturer#9	*Manufacturer#4

```

2019/08/14 23:39:46 Not OK - diffs found

```

```
Usage:
  diffst [flags]
  diffst [command]

Available Commands:
  cmdl      Test case specified on the command line
  help      Help about any command
  sql       Run ad-hoc SQL

Flags:
  -c, --columns           Columnar output
      --config string     config file (default is $HOME/.diffst.yaml)
  -h, --help              help for diffst
  -a, --show-all         Show all results
  -i, --singl-arg string  Run test for single value only
  -s, --source-db string  Source DB - overrides Test Case and Environment
  -b, --source-sql-only   Source SQL Only
  -x, --target-db string  Target DB - overrides Test Case and Environment
  -d, --target-sql-only   Target SQL Only
  -t, --testcase string   Test Case
  -v, --verbose           Run Verbosely producing lots of diagnostic output

Use "diffst [command] --help" for more information about a command.
```

Limitations

- SQL-accessibility
- Limited by DB Drivers

- **Apache Ignite/GridGain:** <https://github.com/amsokol/ignite-go-client>
- **Apache Impala:** <https://github.com/bippio/go-impala>
- **Apache Avatica/Phoenix:** <https://github.com/apache/calcite-avatica-go>
- **AWS Athena:** <https://github.com/segmentio/go-athena>
- **ClickHouse** (uses native TCP interface): <https://github.com/kshvakov/clickhouse>
- **ClickHouse** (uses HTTP API): <https://github.com/mailru/go-clickhouse>
- **CockroachDB:** Use any PostgreSQL driver
- **Couchbase N1QL:** https://github.com/couchbase/go_n1ql
- **DB2 LUW and DB2/Z with DB2-Connect:** <https://bitbucket.org/phiggins/db2cli> (Last updated 2015-08)
- **DB2 LUW** (uses cgo): <https://github.com/asifjalil/cli>
- **DB2 LUW, z/OS, iSeries and Informix:** https://github.com/ibmdb/go_ibm_db
- **Firebird SQL:** <https://github.com/nakagami/firebirdsql>
- **MS ADODB:** <https://github.com/mattn/go-adodb>
- **MS SQL Server** (pure go): <https://github.com/denisenkomp/go-mssqldb>
- **MS SQL Server** (uses cgo): <https://github.com/minus5/gofreetds>
- **MySQL:** <https://github.com/ziutek/mymysql> [*]
- **MySQL:** <https://github.com/go-sql-driver/mysql> [*]
- **ODBC:** <https://bitbucket.org/miquella/mgodbc> (Last updated 2016-02)
- **ODBC:** <https://github.com/alexbrainman/odbc>
- **Oracle:** <https://github.com/mattn/go-oci8>
- **Oracle:** <https://gopkg.in/rana/ora.v4>
- **Oracle:** <https://gopkg.in/goracle.v2>

Source and Target database options - 1

- **QL**: <http://godoc.org/github.com/cznic/ql/driver>
- **Postgres** (pure Go): <https://github.com/lib/pq> [*]
- **Postgres** (uses cgo): <https://github.com/jbarham/gopgsqldriver>
- **Postgres** (pure Go): <https://github.com/jackc/pgx> [**]
- **Presto**: <https://github.com/prestodb/presto-go-client>
- **SAP HANA** (uses cgo): <https://help.sap.com/viewer/0eec0d68141541d1b07893a39944924e/2.0.03/en-US/0ffbe86c9d9f44338441829c6bee15e6.html>
- **SAP HANA** (pure go): <https://github.com/SAP/go-hdb>
- **SAP ASE** (uses cgo): <https://github.com/SAP/go-ase> - package cgo (pure go package planned)
- **Snowflake** (pure Go): <https://github.com/snowflakedb/gosnowflake>
- **SQLite** (uses cgo): <https://github.com/mattn/go-sqlite3> [*]
- **SQLite** (uses cgo): <https://github.com/gwenn/gosqlite> - Supports SQLite dynamic data typing
- **SQLite** (uses cgo): <https://github.com/mxk/go-sqlite>
- **SQLite**: (uses cgo): <https://github.com/rsc/sqlite>
- **SQL over REST**: <https://github.com/adaptant-labs/go-sql-rest-driver>
- **Sybase SQL Anywhere**: <https://github.com/a-palchikov/sqlago>
- **Sybase ASE** (pure go): <https://github.com/thda/tds>
- **Vitess**: <https://godoc.org/vitess.io/vitess/go/vt/vitessdriver>
- **YQL (Yahoo! Query Language)**: <https://github.com/mattn/go-yql>
- **Apache Hive**: <https://github.com/sql-machine-learning/gohive>
- **MaxCompute**: <https://github.com/sql-machine-learning/gomaxcompute>

Source and Target database options - 2

diffst - About

- Developed in Go (golang)
- Go programs compile into statically linked executables for Windows/Linux/Mac/...
- ⇒ No VM required



Advantages of bespoke tool

- Coverage - extends the reach of manual testing
- Bug lifecycle is simpler, faster, better
- CLI, numeric exit status, text input and output supports:
 - Job scheduling
 - Interface with test management tool via API
 - Customizable test suites with scripting
 - Deployability - tester laptop or server

Questions/Comments?