Search-Based Testing of Relational Schema Integrity Constraints Across Multiple Database Management Systems

Matthew Hajduk
Ian Macmillan
Marco Corona
Braden Licastro

Definitions

- Database/ DBMS
- Schema
- Integrity Constraints
  - Foreign Key Constraints
  - Primary Key Constraints
  - Check Constraint
- Search-Based
- Mutation Testing

Motivation

$611 Billion Lost

due to lack of testing of Databases

Contributions

- Search-Based Method for generating data that can satisfy integrity constraints across several DBMSs
- A set of mutation operators for evaluating table data intended to test integrity constraints
- An empirical study with 25 schemas and 3 different DBMSs that compares SchemaAnalyst to DBMonster

Data Generation Preparation

SchemaAnalyst supports three DBMS

- Postgres
- HSQLDB
- SQLite

Database schema must be made DBMS independent, or abstractly represented

Abstract mapping must use universal types.

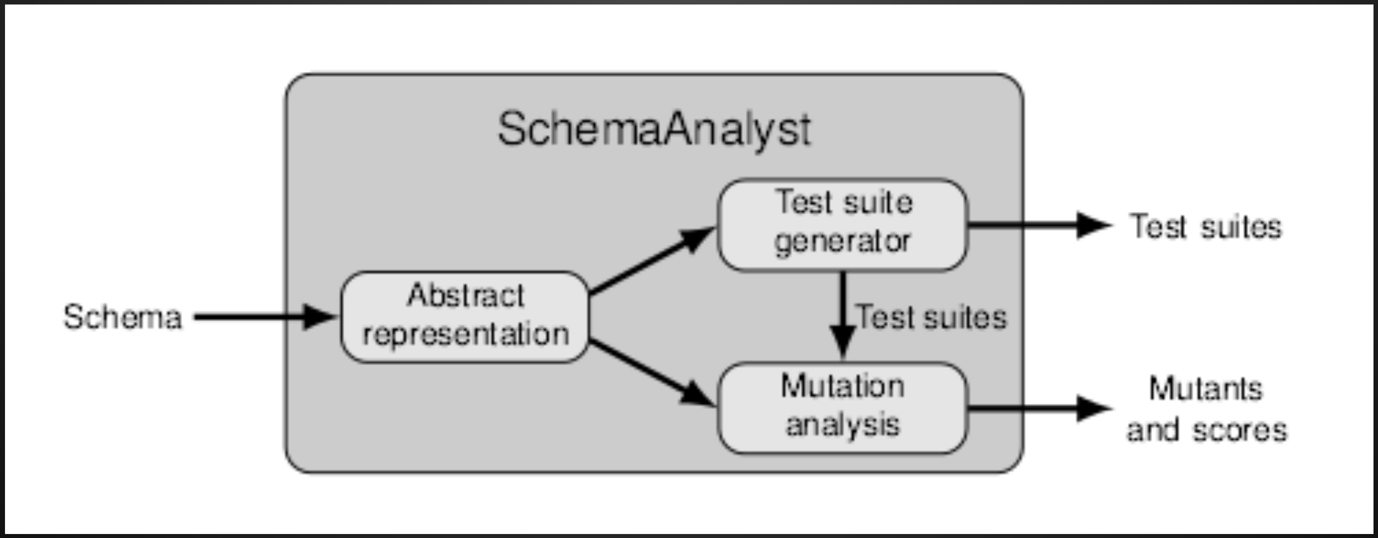
Universal types

- `Boolean`
- `DateTime`
- `Date`
- `Numeric`
- `String`
- `Time`
- `Timestamp`

Generation Algorithm

Goal is to create `INSERT` statements containing data values to make a test suite

- Stage One: Satisfy the schema.
- Stage Two: Violate the schema.



Fitness Function

Purpose is to guide the algorithm to a point of interest -- satisfying the constraint system

Lower the fitness value, closer to the goal it is

Distance from goal is normalized to prevent objectives from becoming too dominant.

Fitness Function

	FLIGHT_ID	SEGMENT_NUMBER	...
1	'UA21'	1	...
2	'UA3750'	1	...
3	'UA22'	2	...

value_dist(a, op, b)

a	b	
NULL	*any*	return 1
any	NULL	return 1
Atomic	Atomic	return $norm(atomic\_dist(a, op, b))$
Compound	Compound	return $norm(compound\_dist(a, op, b))$

atomic_dist(a, op, b)

op	
$=$	if $(\|a - b\| = 0)$ then return 0 else return $\|a - b\| + 1$
$\neq$	if $(\|a - b\| \neq 0)$ then return 0 else return 1
$<$	if $(a - b < 0)$ then return 0 else return $(a - b) + 1$
$\leq$	if $(a - b \leq 0)$ then return 0 else return $(a - b) + 1$
$>$	if $(b - a < 0)$ then return 0 else return $(b - a) + 1$
$\geq$	if $(b - a \leq 0)$ then return 0 else return $(b - a) + 1$

<u>Fitness Result:</u>

Row 1: .516

Row 2: .592

Check Constraints

SchemaAnalyst must also account for:

- Unique constraints
- Foreign Keys
- Not null constraints
- Check constraints
  - Arbitrary conditional
  - ***BETWEEN*** operator
  - ***IN*** operator

Value Modification

Utilizes Korel's *Alternating Variable Method*

- `NULL` statuses flipped
- `BOOLEAN` values are flipped
- `Numeric` or `Timestamp`
  - Values increased
  - Values decreased

Values where fitness improved are saved

Once a fitness of 0 is reached, the AVM is terminated, otherwise process restarts on randomly generated values

Mutation Analysis

Mutation Operators for each Integrity Constraint

- Primary Key
- Unique
- Not Null
- Foreign Keys
- Check constraint

Mutation Operators

Primary Key

Add Column

```
PRIMARY KEY(FLIGHT_ID, SEGMENT_NUMBER,
            ORIGINAL_AIRPORT)
```

Replace Column

```
PRIMARY KEY(SEGMENT_NUMBER, ORIGINAL-AIRPORT)
```

Remove Column

```
PRIMARY KEY(FLIGHT_ID)
```

Mutation Operators

UNIQUE

- Same three forms as Primary Key
- Multiple Unique constraints
- No identical mutants

NOT NULL

- Reverse non-primary key constraint

FOREIGN KEYS

- Removes each foreign key constraint

Mutation Operators

CHECK CONSTRAINT

- Mutant produced one at a time

TOTAL MUTANTS - 56

- **PRIMARY KEY - 31**
- **UNIQUE - 13**
- **NOT NULL - 9**
- **FOREIGN KEY - 2**
- **CHECK - 1**

Empirical Study

- 25 schemas
- 3 Different DBMSs
- Author's Technique vs DBMonster

Case Studies

- Authors selected 25 schemas from:
  - textbooks
  - laboratory assignments
  - online tutorials
  - Postgres examples

- Included a number of real-world applications:
  - Cloc
  - JWhoisServer
  - RiskIt
  - UnixUsage

Table II
CASE STUDY SCHEMAS USED IN THE EMPIRICAL STUDY

Schema	Tables	Columns	Checks	Foreign keys	Not Nulls	Primary keys	Uniques	Total Constraints
BankAccount	2	9	0	1	5	2	0	8
BookTown	23	69	1	0	17	11	0	29
Cloc	2	10	0	0	0	0	0	0
CoffeeOrders	5	20	0	4	9	5	0	18
CustomerOrder	7	32	1	7	27	7	0	42
DellStore	8	52	0	0	36	0	0	36
Employee	1	7	3	0	0	1	0	4
Examination	2	21	6	1	0	2	0	9
Flights	2	13	1	1	6	2	0	10
FrenchTowns	3	14	0	2	13	0	8	23
Inventory	1	4	0	0	0	1	1	2
Iso3166	1	3	0	0	2	1	0	3
JWhoisServer	6	49	0	0	44	6	0	50
NistDML181	2	7	0	1	0	1	0	2
NistDML182	2	32	0	1	0	1	0	2
NistDML183	2	6	0	1	0	0	1	2
NistWeather	2	9	5	0	2	2	0	9
NistXTS748	1	3	1	0	1	0	1	3
NistXTS749	2	7	1	1	3	2	0	7
Person	1	5	1	0	5	1	0	7
Products	3	9	4	2	5	3	0	14
RiskIt	13	56	0	10	15	11	0	36
StudentResidence	2	6	3	1	2	2	0	8
UnixUsage	8	32	0	7	9	7	0	23
Usda	10	67	0	0	30	0	0	30
Total	111	542	27	40	231	68	11	377

Table II, page 6

Case Studies

- Schemas configured to run on 3 DBMSs:
  - Postgres
  - HSQLDB
  - SQLite

- Data generation methods configured to interact with DBMSs through JDBC drivers.

Technique Configuration

- AVM(Author's approach)
  - Allowed for at most 100,000 fitness evaluations
  - Aimed to satisfy schema with 2 rows per table
    - $n_s = 2$
  - Violate schema with 1 row per table
    - $n_v = 1$
- DBMonster
  - Allowed for at most 100,000 data generation attempts
  - Required to generate at least 50 rows per table
  - Used with default configuration

Difficulties Configuring DBMonster

- Does not record database interactions
  - Cobb et al.'s approach
- It's a one-eyed monster
  - Only usable with Postgres
- Instability

Technique Configuration

- AVM able to successfully generated data for all 3 DBMSs
- Randomness
- 30 trials
- Both techniques tested in same environment

Evaluation Metrics

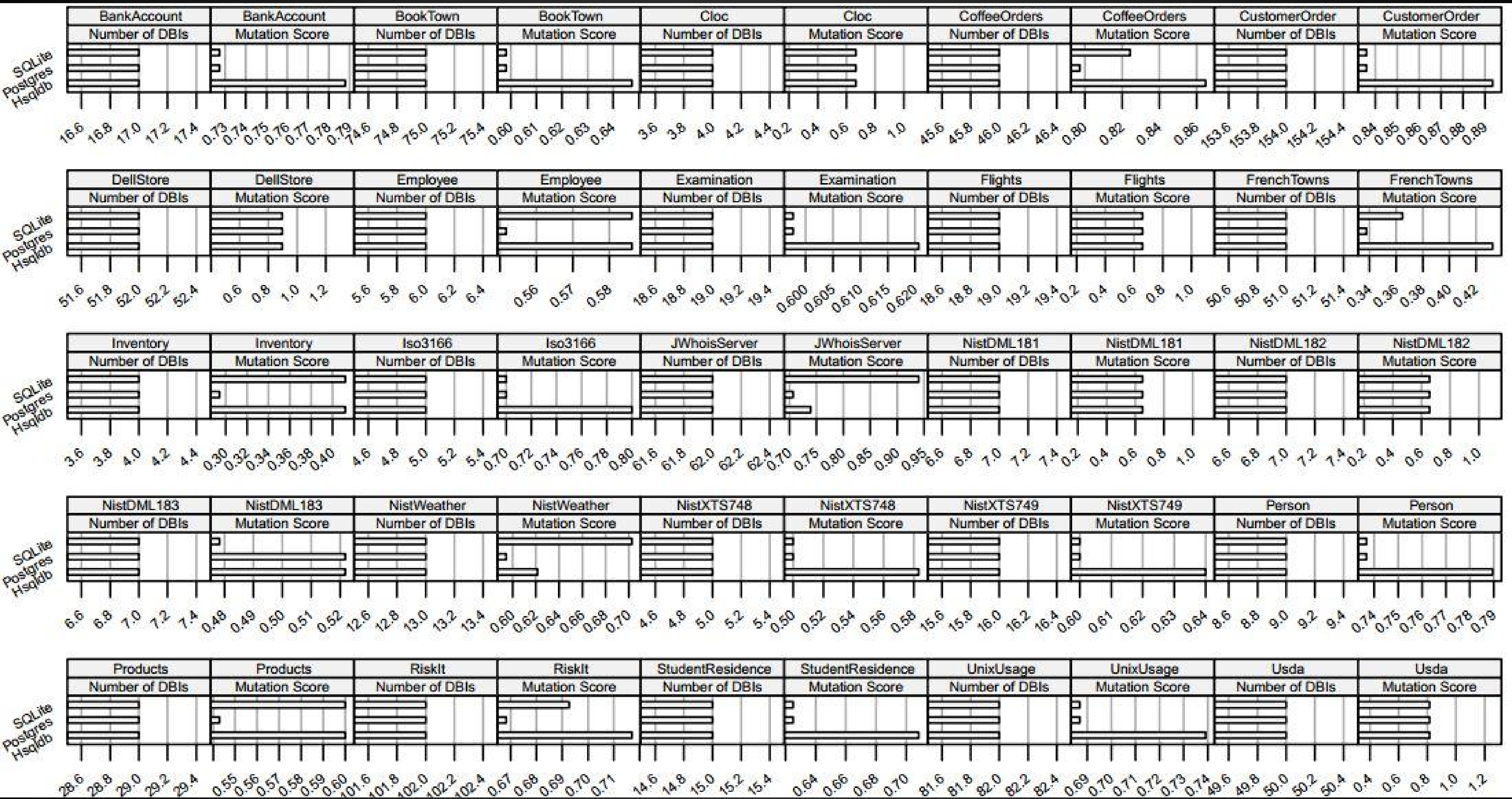
- Measured execution time
- Constraint Coverage
- Mutation Analysis

Mutation Analysis

- Higher-is-better mutation score

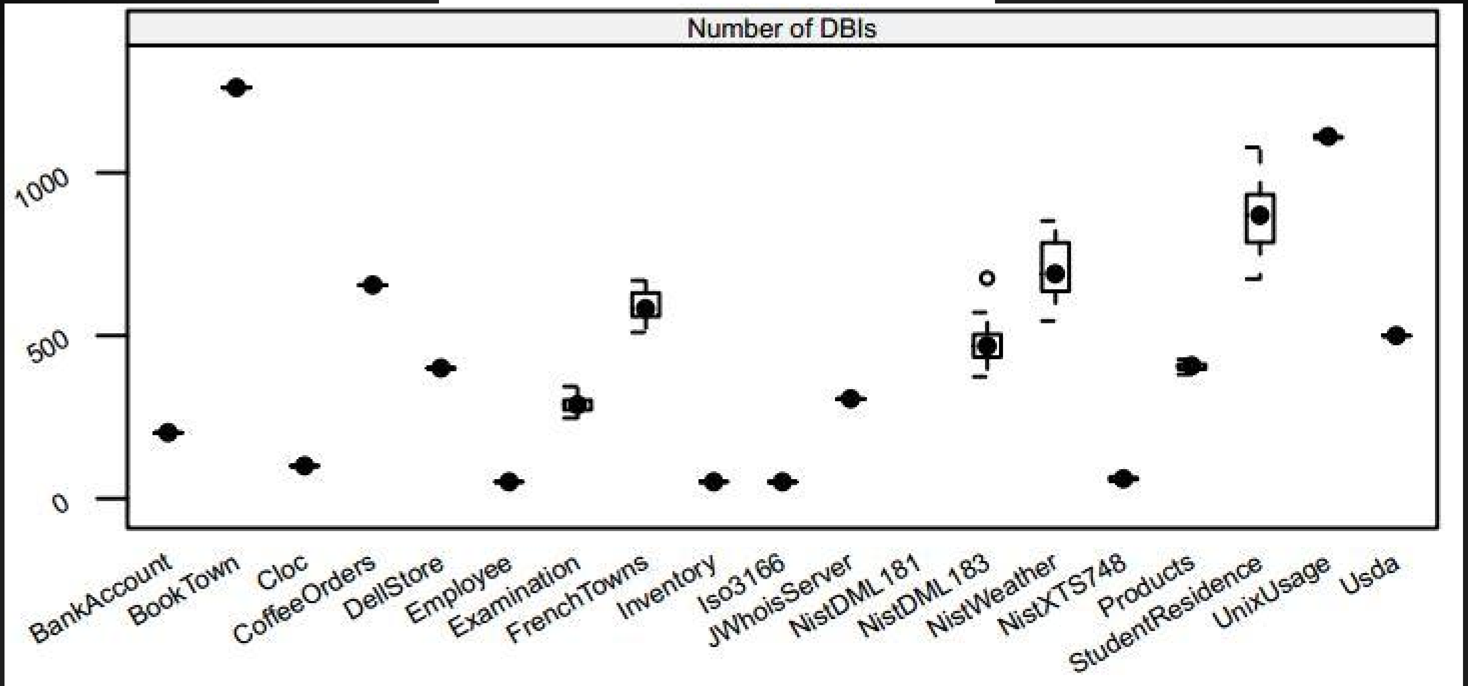
- $M_D = |K \cup Q| / |K \cup N|$
  - D=Set of database interactions
  - Q=Set of quasi-mutants
  - K=Killed Mutants
  - N=Not killed mutants

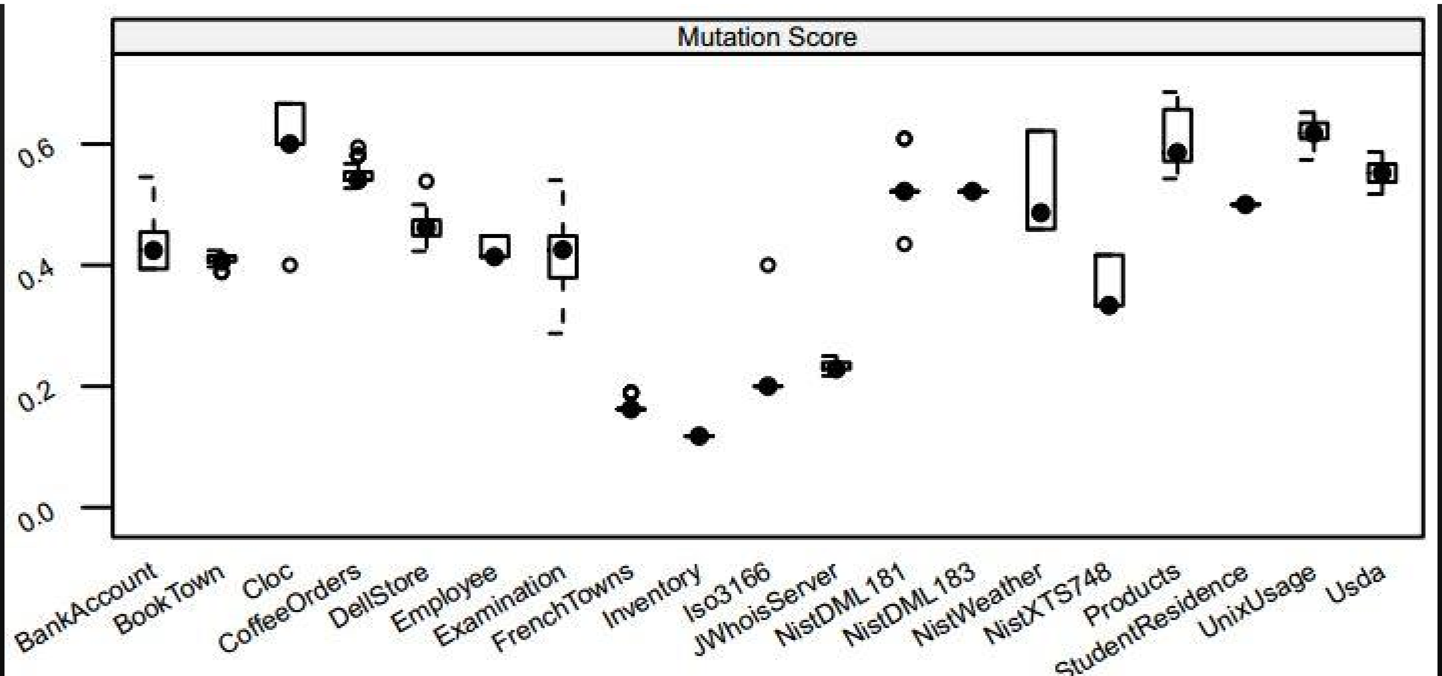
- Iteratively apply each operator to a schema to produce mutants



SQLite
Postgres
Hsqldb
BankAccount Number of DBIs
16.6 16.8 17.0 17.2 17.4
BankAccount Mutation Score
0.73 0.74 0.75 0.76 0.77 0.78 0.79
BookTown Number of DBIs
74.6 74.8 75.0 75.2 75.4
BookTown Mutation Score
0.60 0.61 0.62 0.63 0.64
Cloc Number of DBIs
3.6 3.8 4.0 4.2 4.4
Cloc Mutation Score
0.2 0.4 0.6 0.8 1.0
CoffeeOrders Number of DBIs
45.6 45.8 46.0 46.2 46.4
CoffeeOrders Mutation Score
0.80 0.82 0.84 0.86
CustomerOrder Number of DBIs
153.6 153.8 154.0 154.2 154.4
CustomerOrder Mutation Score
0.84 0.85 0.86 0.87 0.88 0.89
DellStore Number of DBIs
51.6 51.8 52.0 52.2 52.4
DellStore Mutation Score
0.6 0.8 1.0 1.2
Employee Number of DBIs
5.6 5.8 6.0 6.2 6.4
Employee Mutation Score
0.56 0.57 0.58
Examination Number of DBIs
18.6 18.8 19.0 19.2 19.4
Examination Mutation Score
0.600 0.605 0.610 0.615 0.620
Flights Number of DBIs
18.6 18.8 19.0 19.2 19.4
Flights Mutation Score
0.2 0.4 0.6 0.8 1.0
FrenchTowns Number of DBIs
50.6 50.8 51.0 51.2 51.4
FrenchTowns Mutation Score
0.34 0.36 0.38 0.40 0.42
Inventory Number of DBIs
3.6 3.8 4.0 4.2 4.4
Inventory Mutation Score
0.30 0.32 0.34 0.36 0.38 0.40
Iso3166 Number of DBIs
4.6 4.8 5.0 5.2 5.4
Iso3166 Mutation Score
0.70 0.72 0.74 0.76 0.78 0.80
JWhoisServer Number of DBIs
61.6 61.8 62.0 62.2 62.4
JWhoisServer Mutation Score
0.70 0.75 0.80 0.85 0.90 0.95
NistDML181 Number of DBIs
6.6 6.8 7.0 7.2 7.4
NistDML181 Mutation Score
0.2 0.4 0.6 0.8 1.0
NistDML182 Number of DBIs
6.6 6.8 7.0 7.2 7.4
NistDML182 Mutation Score
0.2 0.4 0.6 0.8 1.0
NistDML183 Number of DBIs
6.6 6.8 7.0 7.2 7.4
NistDML183 Mutation Score
0.48 0.49 0.50 0.51 0.52
NistWeather Number of DBIs
12.6 12.8 13.0 13.2 13.4
NistWeather Mutation Score
0.60 0.62 0.64 0.66 0.68 0.70
NistXTS748 Number of DBIs
4.6 4.8 5.0 5.2 5.4
NistXTS748 Mutation Score
0.50 0.52 0.54 0.56 0.58
NistXTS749 Number of DBIs
15.6 15.8 16.0 16.2 16.4
NistXTS749 Mutation Score
0.60 0.61 0.62 0.63 0.64
Person Number of DBIs
8.6 8.8 9.0 9.2 9.4
Person Mutation Score
0.74 0.75 0.76 0.77 0.78 0.79
Products Number of DBIs
28.6 28.8 29.0 29.2 29.4
Products Mutation Score
0.55 0.56 0.57 0.58 0.59 0.60
RiskIt Number of DBIs
101.6 101.8 102.0 102.2 102.4
RiskIt Mutation Score
0.67 0.68 0.69 0.70 0.71
StudentResidence Number of DBIs
14.6 14.8 15.0 15.2 15.4
StudentResidence Mutation Score
0.64 0.66 0.68 0.70
UnixUsage Number of DBIs
81.6 81.8 82.0 82.2 82.4
UnixUsage Mutation Score
0.69 0.70 0.71 0.72 0.73 0.74
Usda Number of DBIs
49.6 49.8 50.0 50.2 50.4
Usda Mutation Score
0.4 0.6 0.8 1.0 1.2

Number of DBIs / Mutation Score





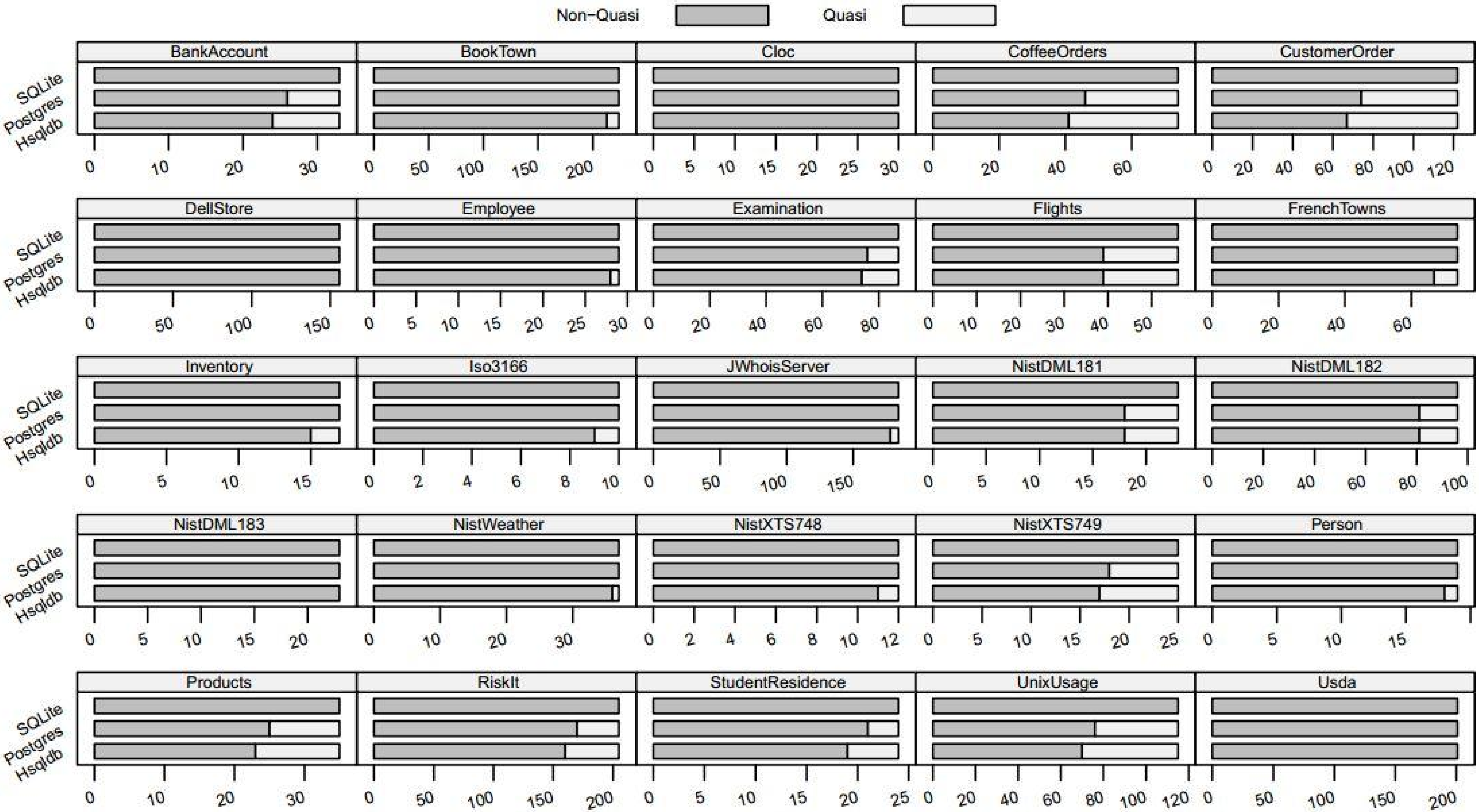

Non−Quasi
Quasi
BankAccount
BookTown
Cloc
CoffeeOrders
CustomerOrder
DellStore
Employee
Examination
Flights
FrenchTowns
Inventory
Iso3166
JWhoisServer
NistDML181
NistDML182
NistDML183
NistWeather
NistXTS748
NistXTS749
Person
Products
RiskIt
StudentResidence
UnixUsage
Usda
SQLite
Postgres
Hsqldb

Table III
CONSTRAINT COVERAGE

Schema	AVM (%)	*DBMonster* (%)
BankAccount	100.0	56.3
BookTown	100.0	51.7
Cloc	*(no constraints defined)*	
CoffeeOrders	100.0	50.0
CustomerOrder	100.0	9.5
DellStore	100.0	50.0
Employee	100.0	55.0
Examination	100.0	72.2
Flights	100.0	70.0
FrenchTowns	100.0	70.0
Inventory	100.0	75.0
Iso3166	100.0	50.0
JWhoisServer	100.0	50.0
NistDML181	100.0	75.0
NistDML182	100.0	50.0
NistDML183	100.0	100.0
NistXTS748	100.0	72.2
NistXTS749	100.0	21.4
NistWeather	100.0	68.7
Person	100.0	50.0
RiskIt	100.0	4.1
Products	96.4	59.3
StudentResidence	100.0	62.5
UnixUsage	97.8	59.3
Usda	100.0	50.0

Efficiency

- Simple Schemas
  - Both take about 5 seconds

- Complex Schemas
  - AVM:
    - *Flights*: 2 seconds
    - NistDML182: 2 seconds
  - DBMonster:
    - *Flights*: 15-30 seconds
    - NistDML182: 634 seconds

Conclusion

- This paper presents SchemaAnalyst
- Presents an Alternating Variable Method (AVM) for generating a test suite
- Presents an Empirical Study where SchemaAnalyst outperforms DBMonster

Future Work

- Employ Constraint Solvers
- Expand for more DBMSs
- Different SQL statements