What Factors Make SQL Test Cases Understandable For Testers? A Human Study of Automatic Test Data Generation Techniques

By **Abdullah Alsharif**, Gregory M. Kapfhammer and Phil McMinn

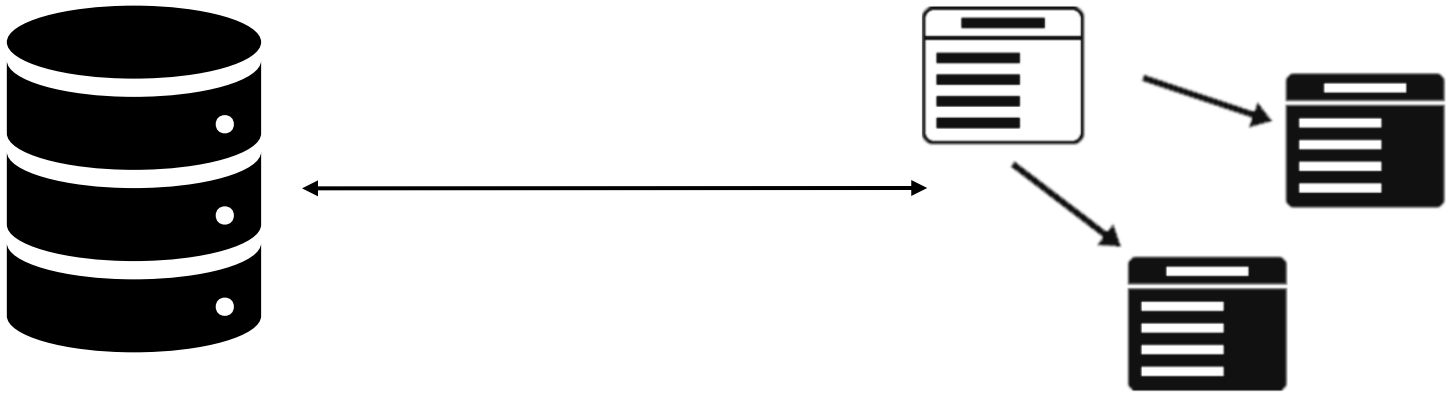
The University Of Sheffield.

ALLEGHENY COLLEGE

DATABASES ARE IMPORTANT
TO EVERY ORGANIZATION

TESTING IS IMPORTANT BUT
IT'S A TEDIOUS TASK

Relational Databases



Database

Schema

Schema can contain many complex integrity constraints

Schema

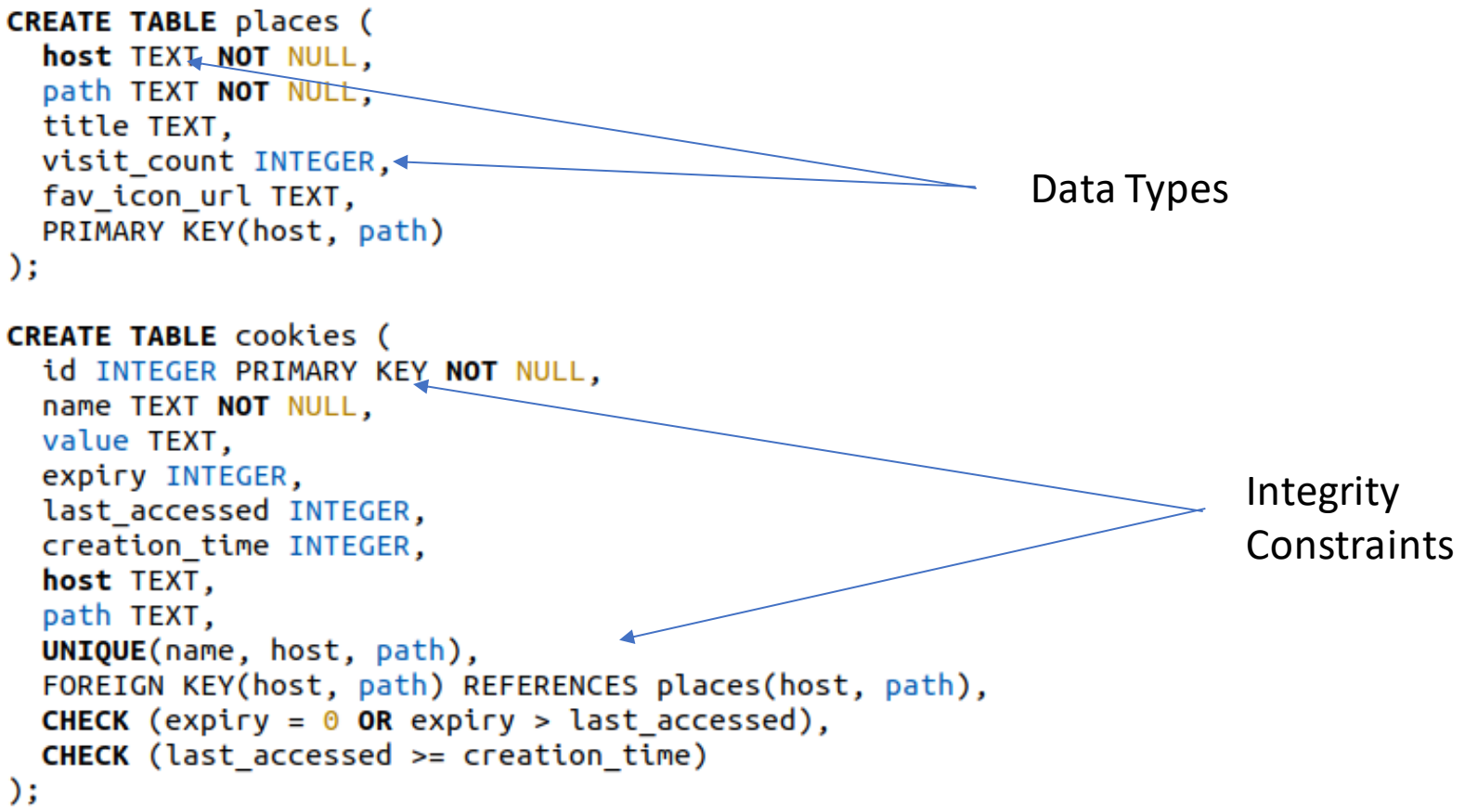
```
CREATE TABLE places (
  host TEXT NOT NULL,
  path TEXT NOT NULL,
  title TEXT,
  visit_count INTEGER,
  fav_icon_url TEXT,
  PRIMARY KEY(host, path)
);

CREATE TABLE cookies (
  id INTEGER PRIMARY KEY NOT NULL,
  name TEXT NOT NULL,
  value TEXT,
  expiry INTEGER,
  last_accessed INTEGER,
  creation_time INTEGER,
  host TEXT,
  path TEXT,
  UNIQUE(name, host, path),
  FOREIGN KEY(host, path) REFERENCES places(host, path),
  CHECK (expiry = 0 OR expiry > last_accessed),
  CHECK (last_accessed >= creation_time)
);
```

Schema

```sql
CREATE TABLE places (
  host TEXT NOT NULL,
  path TEXT NOT NULL,
  title TEXT,
  visit_count INTEGER,
  fav_icon_url TEXT,
  PRIMARY KEY(host, path)
);

CREATE TABLE cookies (
  id INTEGER PRIMARY KEY NOT NULL,
  name TEXT NOT NULL,
  value TEXT,
  expiry INTEGER,
  last_accessed INTEGER,
  creation_time INTEGER,
  host TEXT,
  path TEXT,
  UNIQUE(name, host, path),
  FOREIGN KEY(host, path) REFERENCES places(host, path),
  CHECK (expiry = 0 OR expiry > last_accessed),
  CHECK (last_accessed >= creation_time)
);
```



Generating Tests Automatically

Test Requirement: violate the following constraint

```
UNIQUE(name, host, path),
```

Generating Tests Automatically

Test Requirement: violate the following constraint

```
UNIQUE(name, host, path),
```

AVM-Defaults Generates:

```
1) INSERT INTO places(host, path, title, visit_count, fav_icon_url) VALUES ('', '', '', 0, '')
2) INSERT INTO cookies(id, name, value, expiry, last_accessed, creation_time, host, path) VALUES (0, '', '', 0, 0, 0, '', '')
3) INSERT INTO places(host, path, title,visit_count, fav_icon_url) VALUES ('a', '', '', 0, '')
```

Generating Tests Automatically

Test Requirement: violate the following constraint

```
UNIQUE(name, host, path),
```

AVM-Defaults Generates:

```
1) INSERT INTO places(host, path, title, visit_count, fav_icon_url) VALUES ('', '', '', 0, '')
2) INSERT INTO cookies(id, name, value, expiry, last_accessed, creation_time, host, path) VALUES (0, '', '', 0, 0, 0, '', '')
3) INSERT INTO places(host, path, title,visit_count, fav_icon_url) VALUES ('a', '', '', 0, '')
4) INSERT INTO cookies(id, name, value, expiry, last_accessed, creation_time, host, path) VALUES (1, '', '', 0, 0, 0, '', '')
```

Generating Tests Automatically

Test Requirement: violate the following constraint

```
UNIQUE(name, host, path),
```

AVM-Defaults Generates:

```
1) INSERT INTO places(host, path, title, visit_count, fav_icon_url) VALUES ('', '', '', 0, '')
2) INSERT INTO cookies(id, name, value, expiry, last_accessed, creation_time, host, path) VALUES (0, '', '', 0, 0, 0, '', '')
3) INSERT INTO places(host, path, title,visit_count, fav_icon_url) VALUES ('a', '', '', 0, '')
4) INSERT INTO cookies(id, name, value, expiry, last_accessed, creation_time, host, path) VALUES (1, '', '', 0, 0, 0, '', '')
```

DOMINO-Random Generates:

```
1) INSERT INTO places(host, path, title, visit_count, fav_icon_url) VALUES ('xuksiu', 'fwkjy', 'bmmniu', -53, 'f')
2) INSERT INTO cookies(id, name, value, expiry, last_accessed, creation_time, host, path) VALUES (0, 'iywt', 'ryl', 0, -357, -877, 'xuksiu', 'fwkjy')
3) INSERT INTO places(host, path, title,visit_count, fav_icon_url) VALUES ('lmm', 'j', 'w', 907, NULL)
4) INSERT INTO cookies(id, name, value, expiry, last_accessed, creation_time, host, path) VALUES (131, 'iywt', 'mdofmfl', NULL, NULL, 106, 'xuksiu', 'fwkjy')
```

Are these test understandable?

Test Requirement: violate the following constraint

```
UNIQUE(name, host, path),
```

AVM-Defaults Generates:

```
1) INSERT INTO places(host, path, title, visit_count, fav_icon_url) VALUES ('', '', '', 0, '')
2) INSERT INTO cookies(id, name, value, expiry, last_accessed, creation_time, host, path) VALUES (0, '', '', 0, 0, 0, '', '')
3) INSERT INTO places(host, path, title,visit_count, fav_icon_url) VALUES ('a', '', '', 0, '')
4) INSERT INTO cookies(id, name, value, expiry, last_accessed, creation_time, host, path) VALUES (1, '', '', 0, 0, 0, '', '')
```

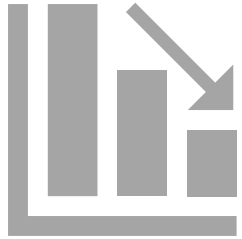
DOMINO-Random Generates:

```
1) INSERT INTO places(host, path, title, visit_count, fav_icon_url) VALUES ('xuksiu', 'fwkjy', 'bmmniu', -53, 'f')
2) INSERT INTO cookies(id, name, value, expiry, last_accessed, creation_time, host, path) VALUES (0, 'iywt', 'ryl', 0, -357, -877, 'xuksiu', 'fwkjy')
3) INSERT INTO places(host, path, title,visit_count, fav_icon_url) VALUES ('lmm', 'j', 'w', 907, NULL)
4) INSERT INTO cookies(id, name, value, expiry, last_accessed, creation_time, host, path) VALUES (131, 'iywt', 'mdofmfl', NULL, NULL, 106, 'xuksiu', 'fwkjy')
```

The Human Oracle Cost

Qualitative Cost

Associated with the level of comprehension required to evaluate the behavior of the test



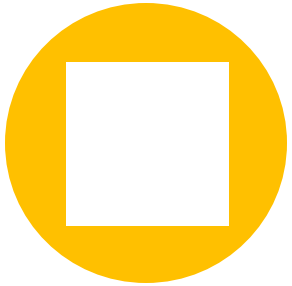
Quantitative Cost

Associated with the test suite size and the time a human takes to evaluate each test case manually

Prior Work

Created more readable values

Created more readable variables



Automated vs Manual tests

No test
comprehension factors identified

Methodology – Current Generators

```
CREATE TABLE places (
  host TEXT NOT NULL,
  path TEXT NOT NULL,
  title TEXT,
  visit_count INTEGER,
  fav_icon_url TEXT,
  PRIMARY KEY(host, path)
);
```

Generator	host	path	title	visit_count	fav_icon_url
AVM-Defaults	''	''	''	0	''
DOMINO-RND	'hctgp'	''	'ra'	0	'kt'

Methodology – Readable Variant Generators

```
CREATE TABLE places (
  host TEXT NOT NULL,
  path TEXT NOT NULL,
  title TEXT,
  visit_count INTEGER,
  fav_icon_url TEXT,
  PRIMARY KEY(host, path)
);
```

Generator	host	path	title	visit_count	fav_icon_url
AVM-Defaults	''	''	''	0	''
DOMINO-RND	'hctgp'	''	'ra'	0	'kt'
AVM-LM	'Thino'	'jongo'	'jesed'	0	'Zesth'

Methodology – Readable Variant Generators

```
CREATE TABLE places (
  host TEXT NOT NULL,
  path TEXT NOT NULL,
  title TEXT,
  visit_count INTEGER,
  fav_icon_url TEXT,
  PRIMARY KEY(host, path)
);
```

Generator	host	path	title	visit_count	fav_icon_url
AVM-Defaults	''	''	''	0	''
DOMINO-RND	'hctgp'	''	'ra'	0	'kt'
AVM-LM	'Thino'	'jongo'	'jesed'	0	'Zesth'
DOMINO-COL	'host_0'	'path_1'	'title_2'	3	'fav_icon_url_4'

Methodology – Readable Variant Generators

```
CREATE TABLE places (
  host TEXT NOT NULL,
  path TEXT NOT NULL,
  title TEXT,
  visit_count INTEGER,
  fav_icon_url TEXT,
  PRIMARY KEY(host, path)
);
```

Generator	host	path	title	visit_count	fav_icon_url
AVM-Defaults	''	''	''	0	''
DOMINO-RND	'hctgp'	''	'ra'	0	'kt'
AVM-LM	'Thino'	'jongo'	'jesed'	0	'Zesth'
DOMINO-COL	'host_0'	'path_1'	'title_2'	3	'fav_icon_url_4'
DOMINO-READ	'sidekick'	'badly'	'numbers'	758	'good'

Methodology – Two Case Studies

```
CREATE TABLE Station (
    ID INTEGER PRIMARY KEY,
    CITY CHAR(20),
    STATE CHAR(2),
    LAT_N INTEGER NOT NULL,
    LONG_W INTEGER NOT NULL,
    CHECK (LAT_N BETWEEN 0 and 90),
    CHECK (LONG_W BETWEEN 180 AND -180)
);

CREATE TABLE Stats (
   ID INTEGER REFERENCES STATION(ID),
   MONTH INTEGER NOT NULL,
   TEMP_F INTEGER NOT NULL,
   RAIN_I INTEGER NOT NULL,
   CHECK (MONTH BETWEEN 1 AND 12),
   CHECK (TEMP_F BETWEEN 80 AND 150),
   CHECK (RAIN_I BETWEEN 0 AND 100),
   PRIMARY KEY (ID, MONTH)
);
```

NistWeather Schema

```
CREATE TABLE places (
  host TEXT NOT NULL,
  path TEXT NOT NULL,
  title TEXT,
  visit_count INTEGER,
  fav_icon_url TEXT,
  PRIMARY KEY(host, path)
);

CREATE TABLE cookies (
  id INTEGER PRIMARY KEY NOT NULL,
  name TEXT NOT NULL,
  value TEXT,
  expiry INTEGER,
  last_accessed INTEGER,
  creation_time INTEGER,
  host TEXT,
  path TEXT,
  UNIQUE(name, host, path),
  FOREIGN KEY(host, path) REFERENCES places(host, path),
  CHECK (expiry = 0 OR expiry > last_accessed),
  CHECK (last_accessed >= creation_time)
);
```

BrowserCookies Schema

Methodology – Survey/Questionnaire

Schema

Question (18/18)

The following question is based on the below schema:

```
CREATE TABLE Station  (
   ID INTEGER PRIMARY KEY,
   CITY CHAR(20),
   STATE CHAR(2),
   LAT_N INTEGER NOT NULL CHECK (LAT_N BETWEEN 0 and 90),
   LONG_W INTEGER NOT NULL CHECK (LONG_W BETWEEN SYMMETRIC 180 AND -180)
);
CREATE TABLE Stats (
  ID INTEGER REFERENCES STATION(ID),
  MONTH INTEGER NOT NULL CHECK (MONTH BETWEEN 1 AND 12),
  TEMP_F INTEGER NOT NULL CHECK (TEMP_F BETWEEN 80 AND 150),
  RAIN_I INTEGER NOT NULL CHECK (RAIN_I BETWEEN 0 AND 100),
  PRIMARY KEY (ID, MONTH)
);
```

The database is initially empty and the following INSERT statements are sequential. If the INSERT statement is successfully accepted, the data will be inserted into the database. The success/failure of the following INSERT statements then depends on this new database state.

```
1. INSERT INTO "Station"("ID", "CITY", "STATE", "LAT_N", "LONG_W") VALUES (0, '', '', 0, 0)
2. INSERT INTO "Stats"("ID", "MONTH", "TEMP_F", "RAIN_I") VALUES (0, 0, 127, 0)
```

Which of the above INSERT statements, if any, will be rejected first?

- ○ INSERT 1
- ○ INSERT 2
- ○ None of them
- ○ I do not know

Submit

Methodology – Survey/Questionnaire

Schema

Test INSERTs

Question (18/18)

The following question is based on the below schema:

```
CREATE TABLE Station  (
   ID INTEGER PRIMARY KEY,
   CITY CHAR(20),
   STATE CHAR(2),
   LAT_N INTEGER NOT NULL CHECK (LAT_N BETWEEN 0 and 90),
   LONG_W INTEGER NOT NULL CHECK (LONG_W BETWEEN SYMMETRIC 180 AND -180)
);
CREATE TABLE Stats (
  ID INTEGER REFERENCES STATION(ID),
  MONTH INTEGER NOT NULL CHECK (MONTH BETWEEN 1 AND 12),
  TEMP_F INTEGER NOT NULL CHECK (TEMP_F BETWEEN 80 AND 150),
  RAIN_I INTEGER NOT NULL CHECK (RAIN_I BETWEEN 0 AND 100),
  PRIMARY KEY (ID, MONTH)
);
```

The database is initially empty and the following INSERT statements are sequential. If the INSERT statement is successfully accepted, the data will be inserted into the database. The success/failure of the following INSERT statements then depends on this new database state.

```
1. INSERT INTO "Station"("ID", "CITY", "STATE", "LAT_N", "LONG_W") VALUES (0, '', '', 0, 0)
2. INSERT INTO "Stats"("ID", "MONTH", "TEMP_F", "RAIN_I") VALUES (0, 0, 127, 0)
```

Which of the above INSERT statements, if any, will be rejected first?

- ○ INSERT 1
- ○ INSERT 2
- ○ None of them
- ○ I do not know

Submit

Methodology – Survey/Questionnaire

Schema

Test INSERTs

The Human Oracle: Which INSERT will be rejected by the DBMS?

Question (18/18)

The following question is based on the below schema:

```
CREATE TABLE Station  (
   ID INTEGER PRIMARY KEY,
   CITY CHAR(20),
   STATE CHAR(2),
   LAT_N INTEGER NOT NULL CHECK (LAT_N BETWEEN 0 and 90),
   LONG_W INTEGER NOT NULL CHECK (LONG_W BETWEEN SYMMETRIC 180 AND -180)
);
CREATE TABLE Stats (
  ID INTEGER REFERENCES STATION(ID),
  MONTH INTEGER NOT NULL CHECK (MONTH BETWEEN 1 AND 12),
  TEMP_F INTEGER NOT NULL CHECK (TEMP_F BETWEEN 80 AND 150),
  RAIN_I INTEGER NOT NULL CHECK (RAIN_I BETWEEN 0 AND 100),
  PRIMARY KEY (ID, MONTH)
);
```

The database is initially empty and the following INSERT statements are sequential. If the INSERT statement is successfully accepted, the data will be inserted into the database. The success/failure of the following INSERT statements then depends on this new database state.

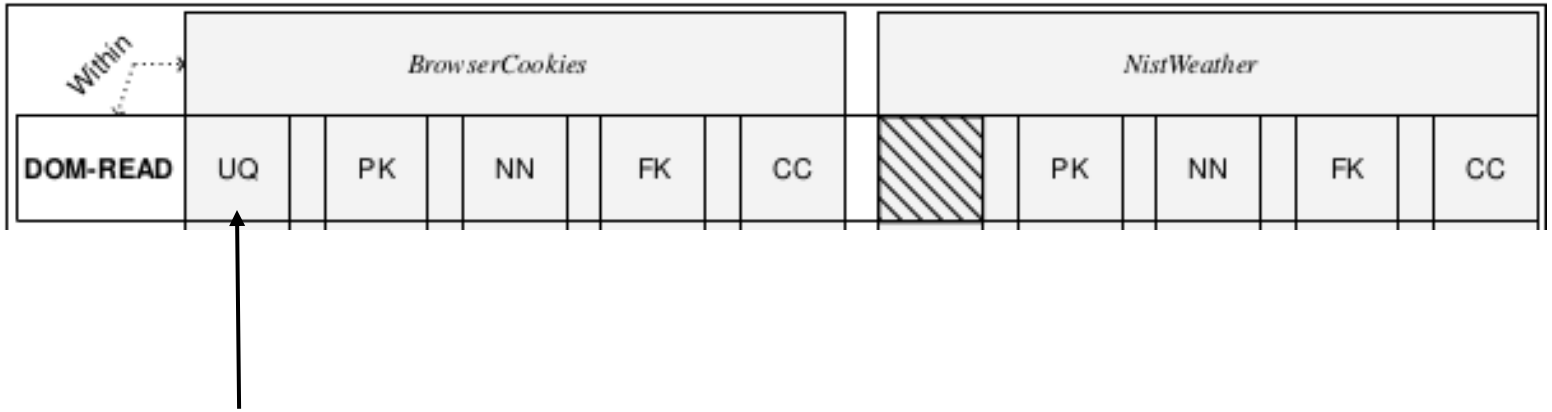
```
1. INSERT INTO "Station"("ID", "CITY", "STATE", "LAT_N", "LONG_W") VALUES (0, '', '', 0, 0)
2. INSERT INTO "Stats"("ID", "MONTH", "TEMP_F", "RAIN_I") VALUES (0, 0, 127, 0)
```

Which of the above INSERT statements, if any, will be rejected first?

- ○ INSERT 1
- ○ INSERT 2
- ○ None of them
- ○ I do not know

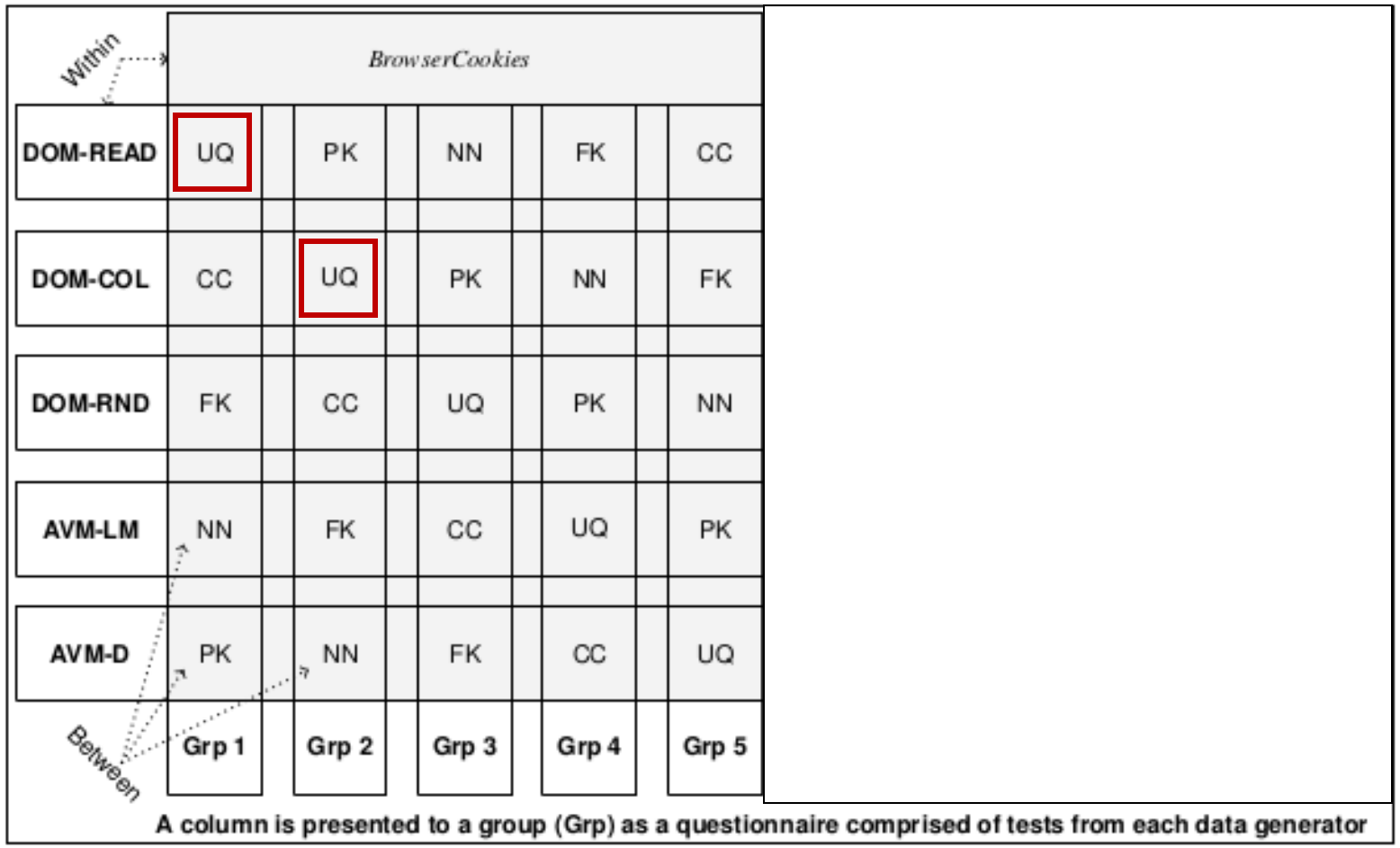
Submit

Methodology – Participant Assignments

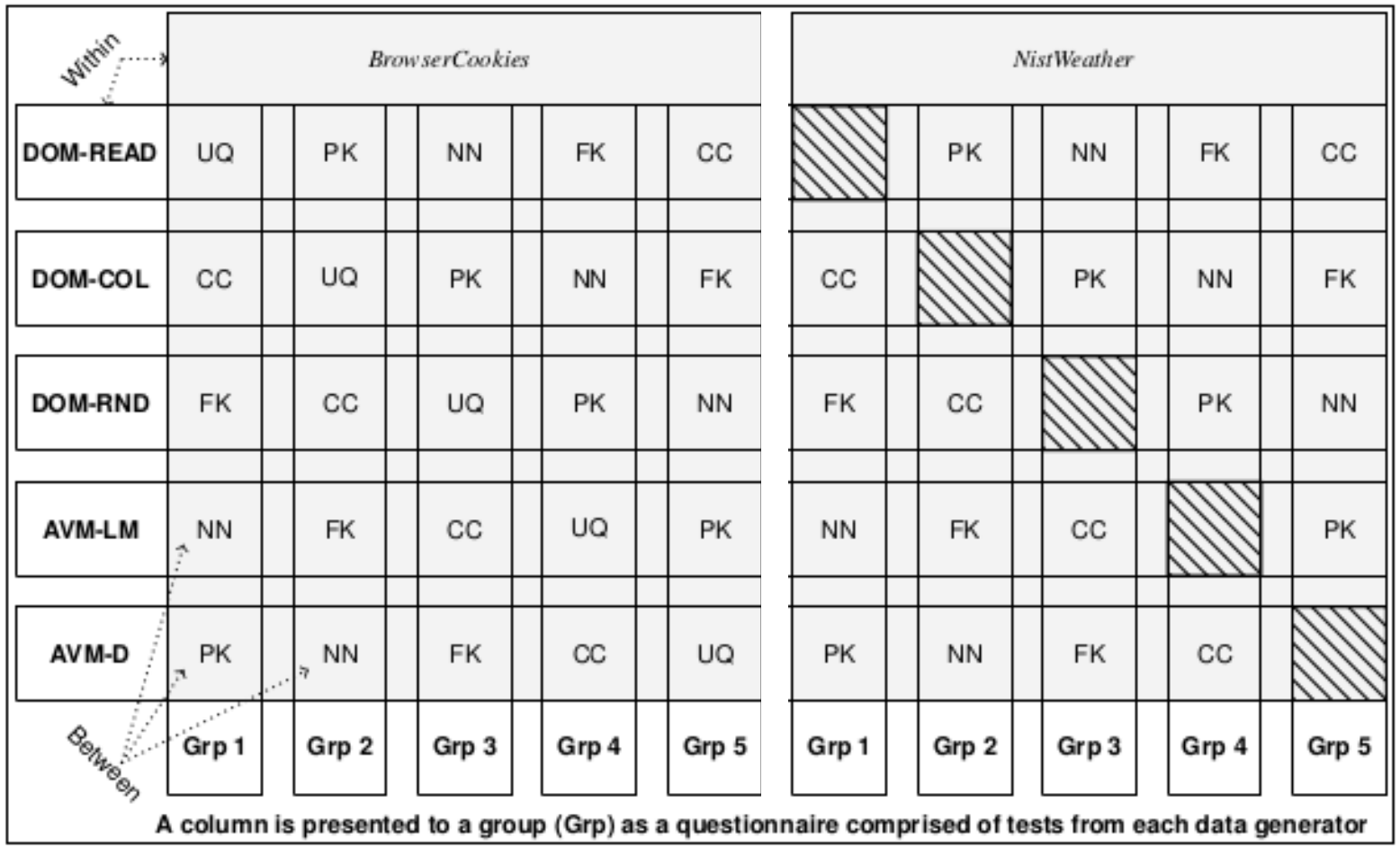


Each integrity constraint has two test case – a violation and a satisfaction

Methodology – Participant Assignments



Methodology – Participant Assignments



Test cases were randomized for each participant in the group

Methodology – Human Study

SILENT STUDY - 25 PARTICIPANTS

THINK ALOUD STUDY – 6 PARTICIPANTS

Methodology – The Think-Aloud Study

- 5 participants with only prompting with a "why?"
- A 6th participant that is an "experienced industry engineer" to corroborate the other 5 participant's comments

Research Questions

RQ1: Success Rate in Comprehending the Test Cases

How successful are testers at correctly comprehending the behavior of schema test cases generated by automated techniques?

RQ2: Factors Involved in Test Case Comprehension

What are the factors of automatically generated SQL INSERT statements that make them easy for testers to understand?

RQ 1 Success Rate – The Silent Study Results

Technique	Correct Responses	Incorrect Responses	Score	Ranking
AVM-DEFAULTs	76	12	84%	1
DOMINO-COL	67	23	74%	2
AVM-LM	65	25	72%	= 3
DOMINO-READ	65	25	72%	= 3
DOMINO-RANDOM	55	35	61%	5

- In conclusion, we observed that AVM-Default is the most easily comprehended
- In contrast, the most difficult to comprehend is DOMINO-RANDOM
- The remaining techniques fall in between these two extremes

What are the factors that contributed to this success rate?

"the NOT NULL constraints are the easiest to spot"

Default Values can show the "differences and similarities between INSERTs"

Default Values can help "to skip over to get to the important data"

"the NOT NULL constraints are the easiest to spot"

Default Values can show the "differences and similarities between INSERTs"

Default Values can help "to skip over to get to the important data"

- It is Easy to Identify When NULL Violates NOT NULL Constraints
- Empty Strings Look Strange, But They Are Helpful

"CHECK constraint should be a NOT NULL by default"

"the path [a FOREIGN KEY] is NULL which is not going to work"

NULLs are confusing with Foreign Keys and CHECK Constraints

"CHECK constraint should be a NOT NULL by default"

"the path [a FOREIGN KEY] is NULL which is not going to work"

NULLs are confusing with Foreign Keys and CHECK Constraints

Negative numbers "takes more time to do mental arithmetic"

Negative numbers are "not realistic"

Negative Numbers Require More Comprehension Effort

"CHECK constraint should be a NOT NULL by default"

"the path [a FOREIGN KEY] is NULL which is not going to work"

NULLs are confusing with Foreign Keys and CHECK Constraints

Negative numbers "takes more time to do mental arithmetic"

Negative numbers are "not realistic"

Negative Numbers Require More Comprehension Effort

Random string are "garbage data"

Random strings "are horrible, they are more distinct"

Random Strings Require More Comprehension Effort

RQ2 Factors – Think Aloud Study Results

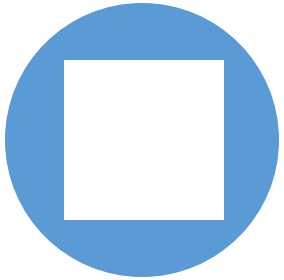
- Participants raised issues concerning the use of NULL, suggesting its judicious use in test data generation
- Positive comments about default values and readable strings
- Dislike of negative numbers and random strings

Conclusion and Recommendations

NULLs are confusing for human testers

Do not use negative numbers as they require testers to think harder

Use simple repetitions for unimportant test values



Use human readable strings values rather than random strings