The University Of Sheffield.

Mutation 2013

ALLEGHENY COLLEGE
MEADVILLE, PENNSYLVANIA

Efficient Mutation Analysis of Relational Database Structure Using Mutant Schemata and Parallelisation

Chris J. Wright
Gregory M. Kapfhammer
Phil McMinn

Relational Database Management Systems

(RDBMS)

SQLite

HyperSQL

Apache Derby

PostgreSQL

Relational Database Management Systems

(RDBMS)

MySQL®

Microsoft® SQL Server®

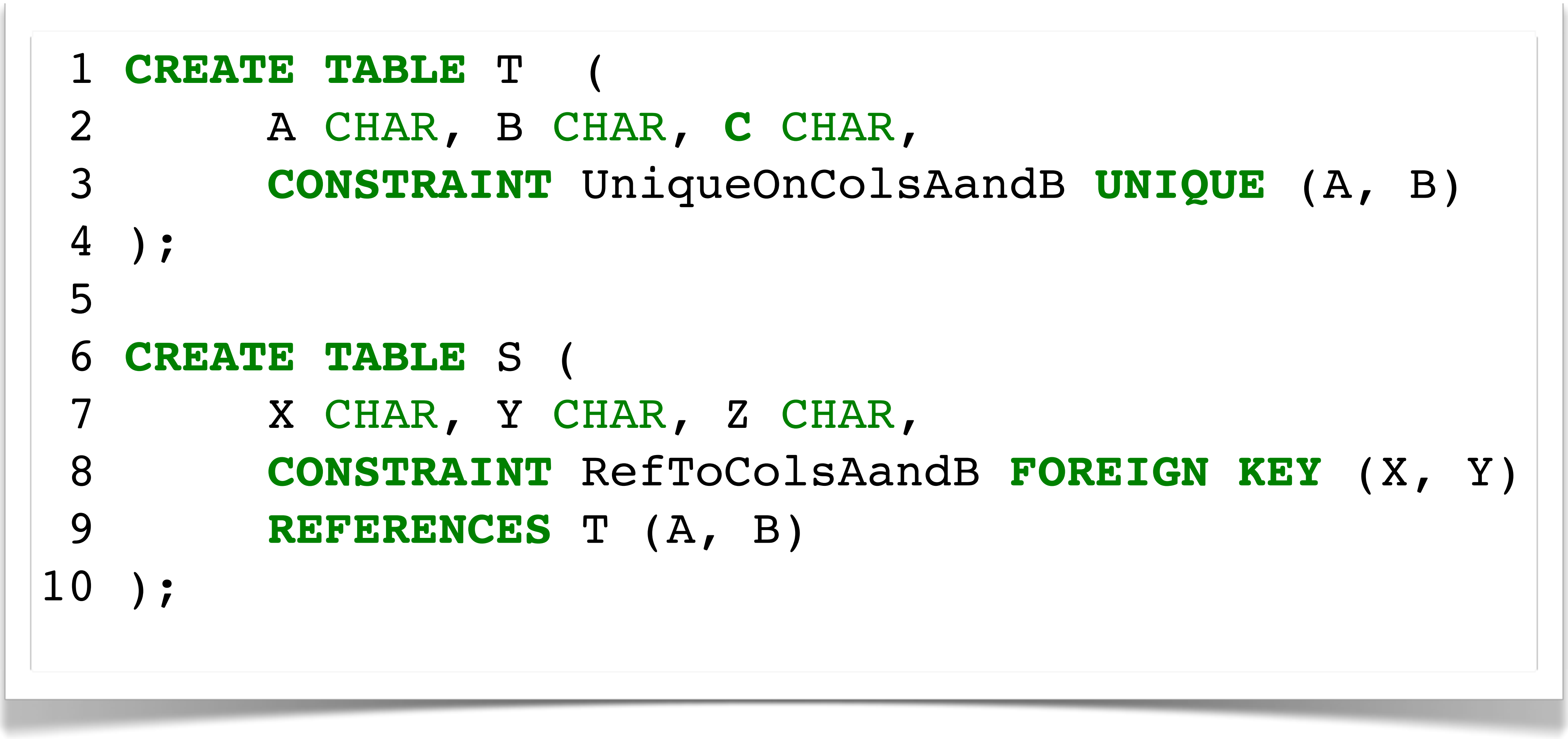
ORACLE® DATABASE

IBM DB2

Many different RDBMSs...

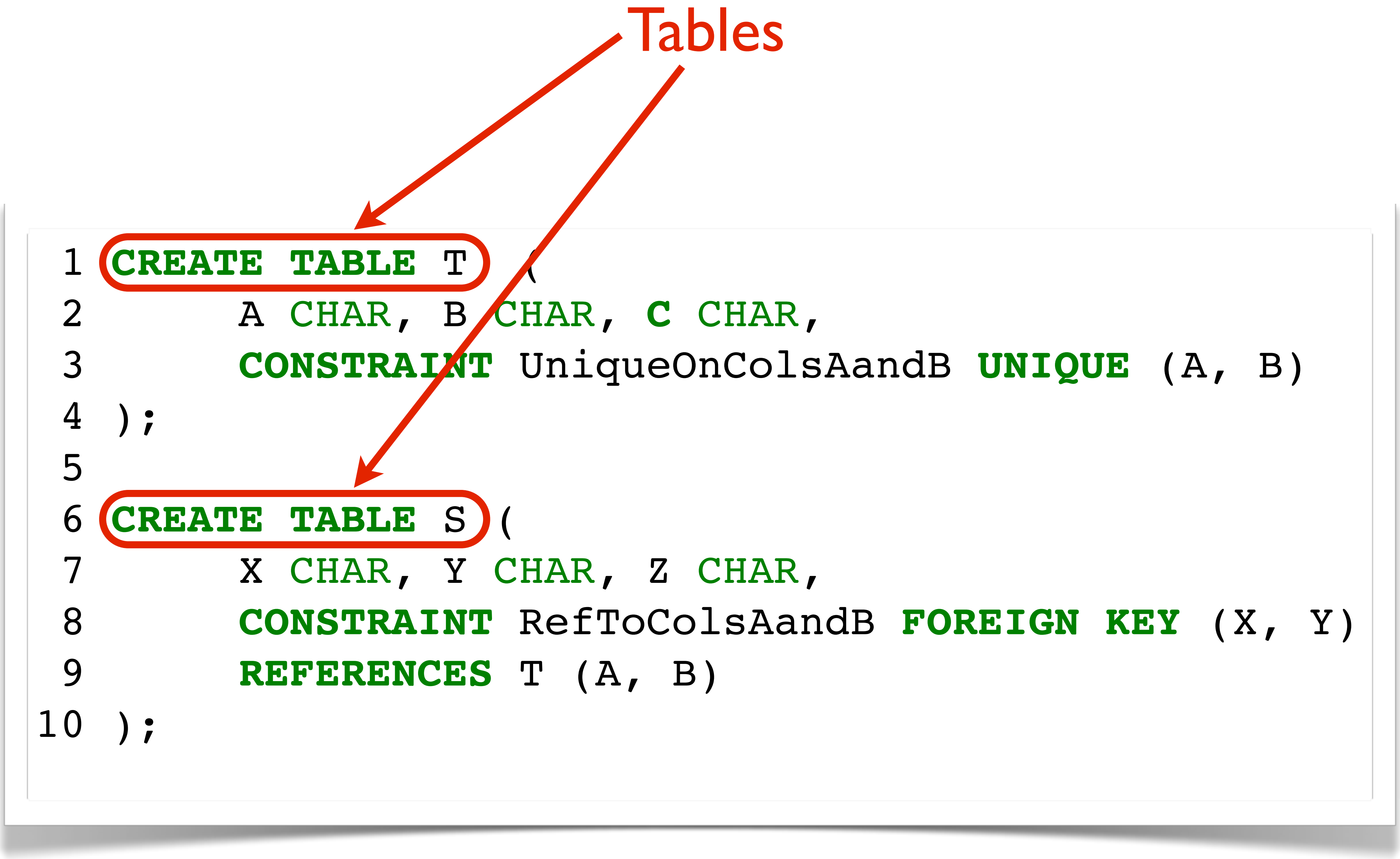
Many different RDBMSs...

...same specification of structure



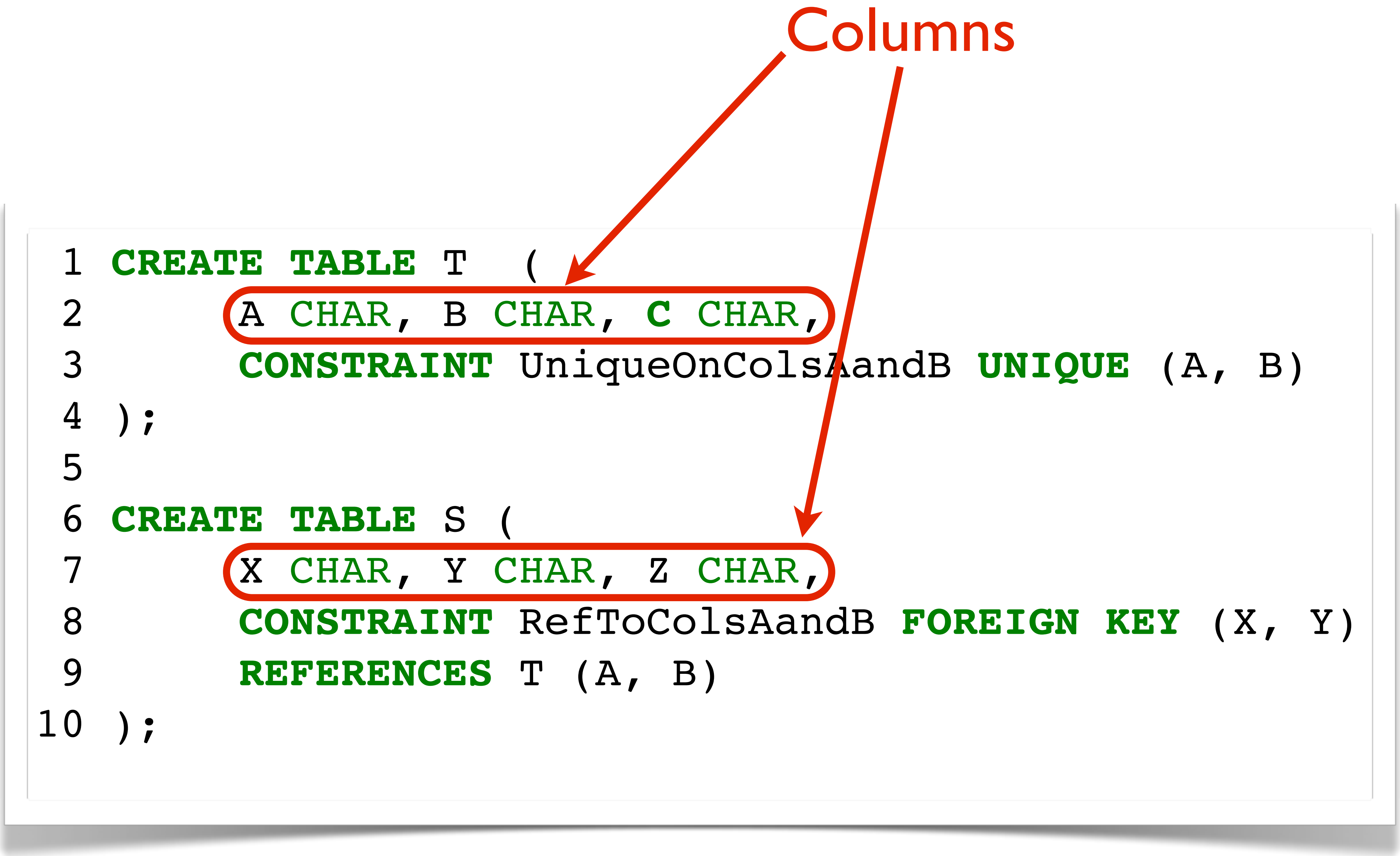
```sql
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```

Database Schema



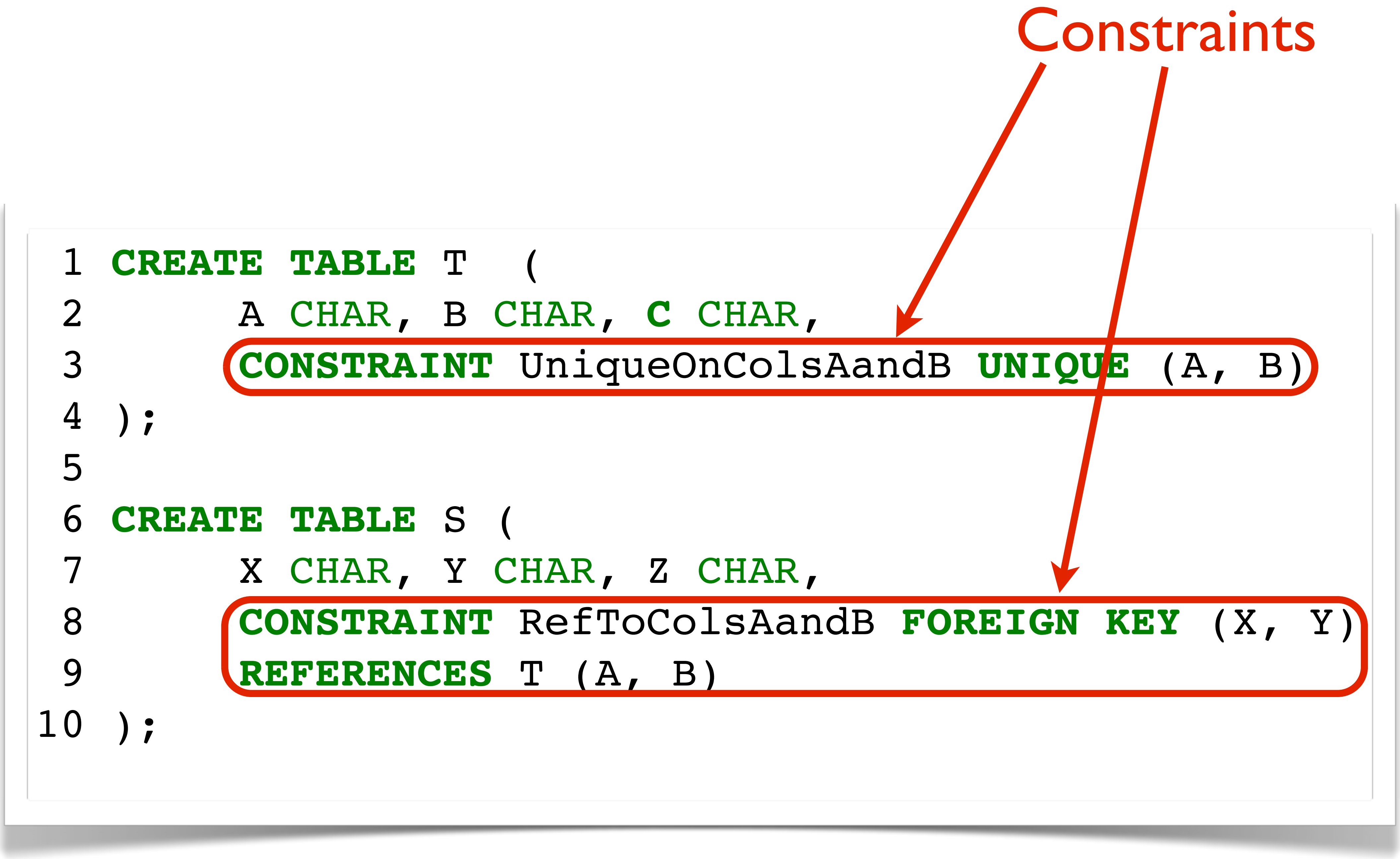
```sql
CREATE TABLE T (
        A CHAR, B CHAR, C CHAR,
        CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
        X CHAR, Y CHAR, Z CHAR,
        CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
        REFERENCES T (A, B)
);
```

Database Schema



```sql
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```

Database Schema

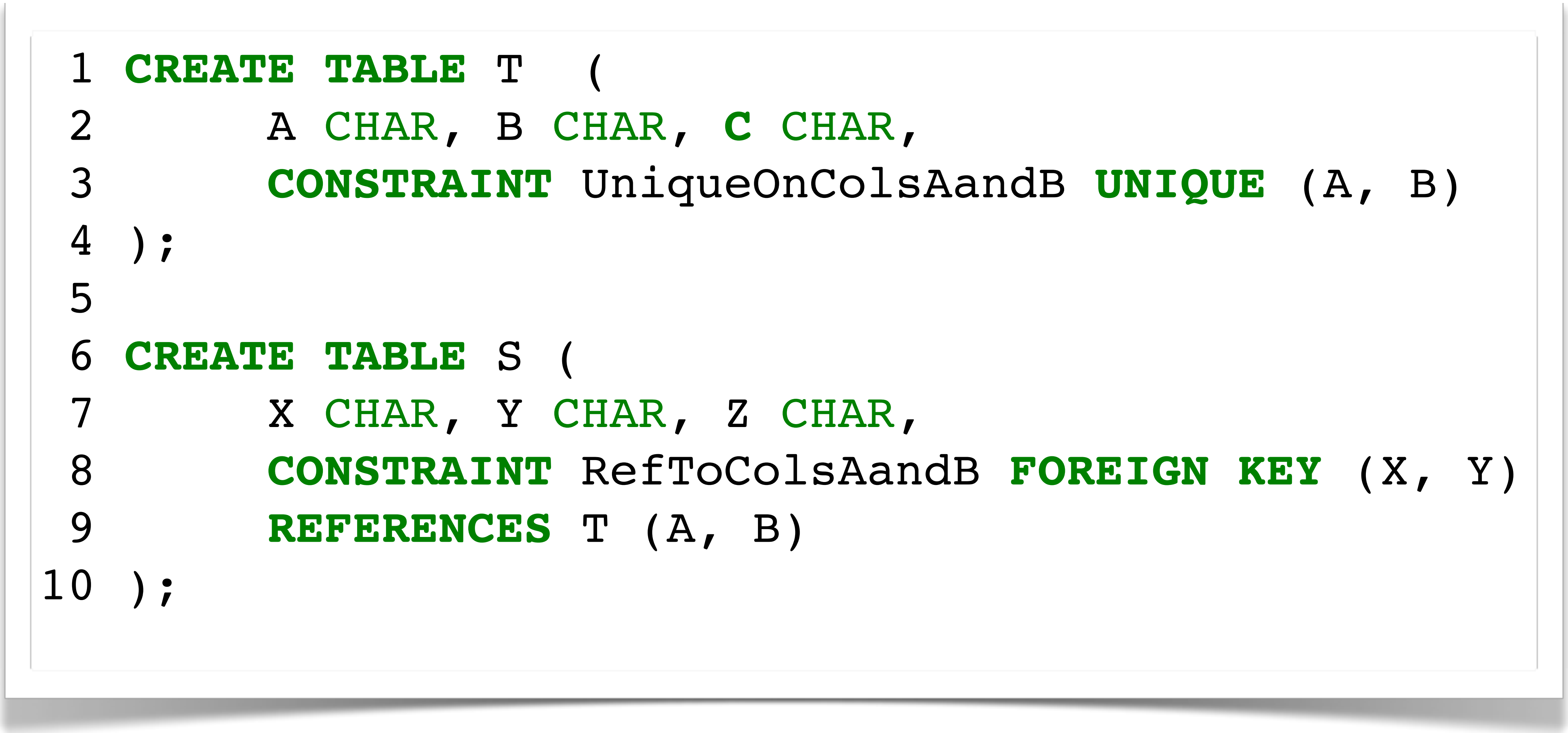


Constraints

```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```

Database Schema

How do we know this is correct?



```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```

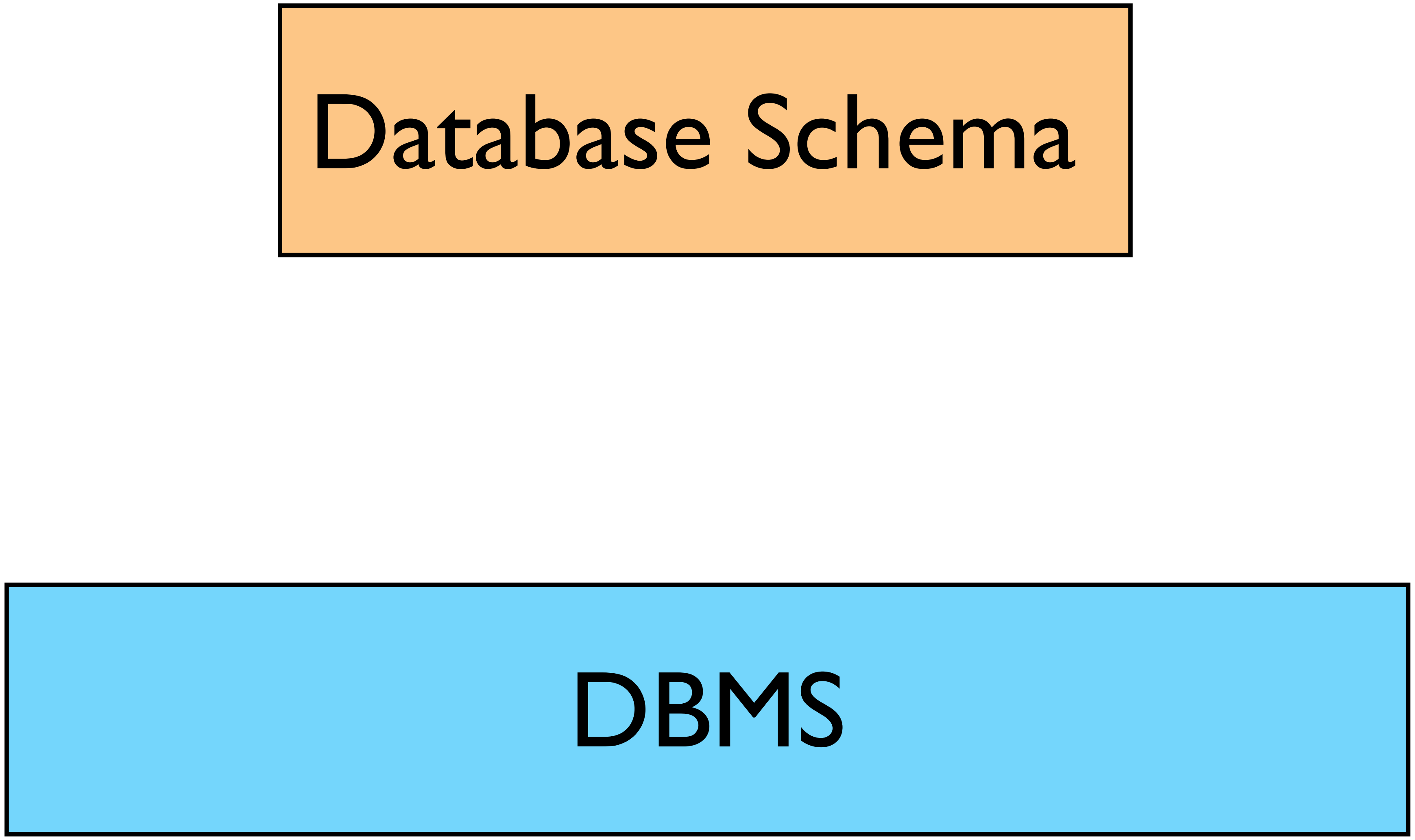
Database Schema

Why Test Database Structure?

Why Test Database Structure?

Database Schema

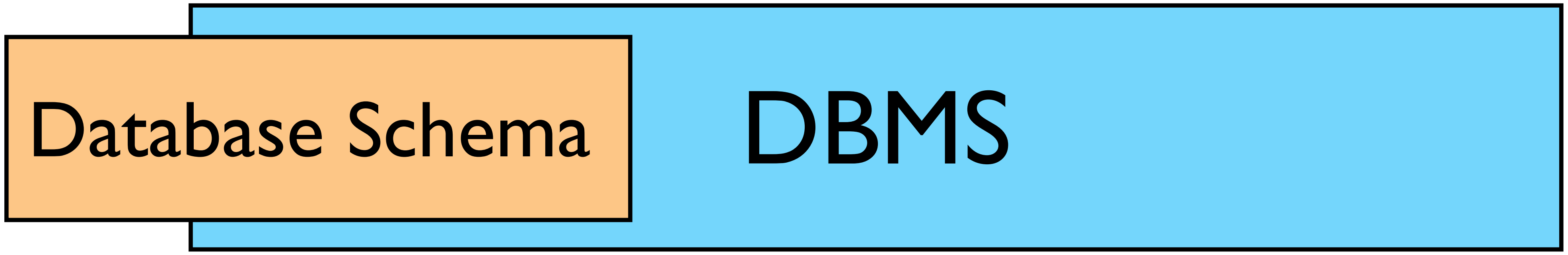
Why Test Database Structure?



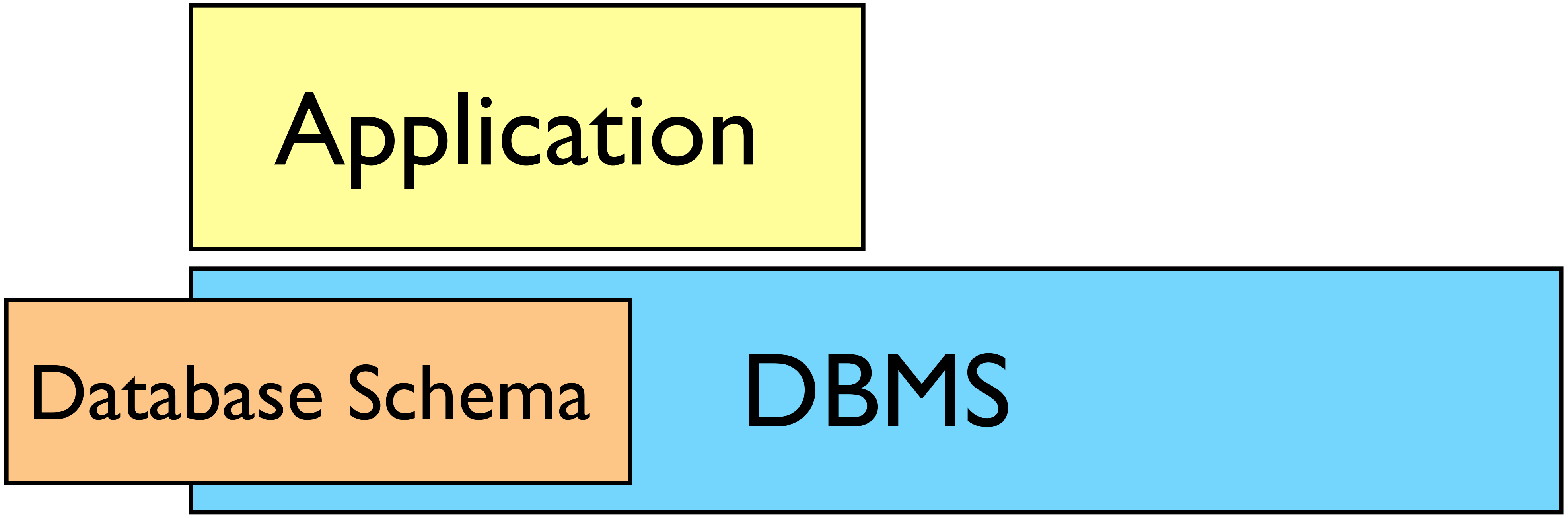
Database Schema

DBMS

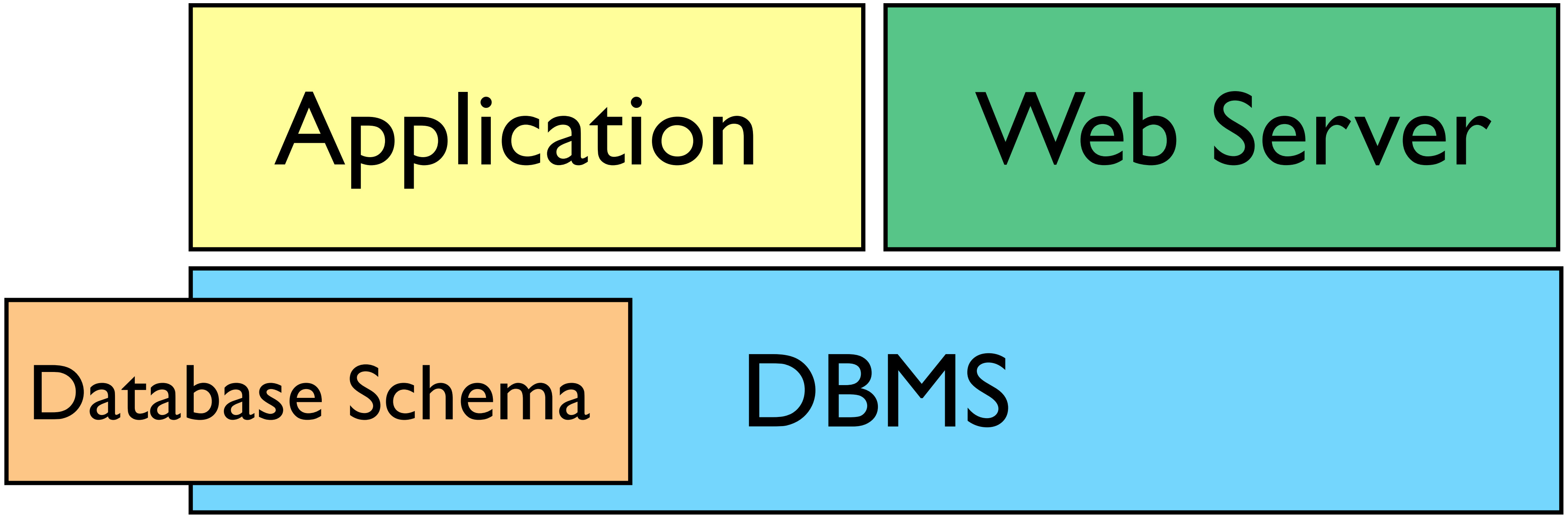
Why Test Database Structure?



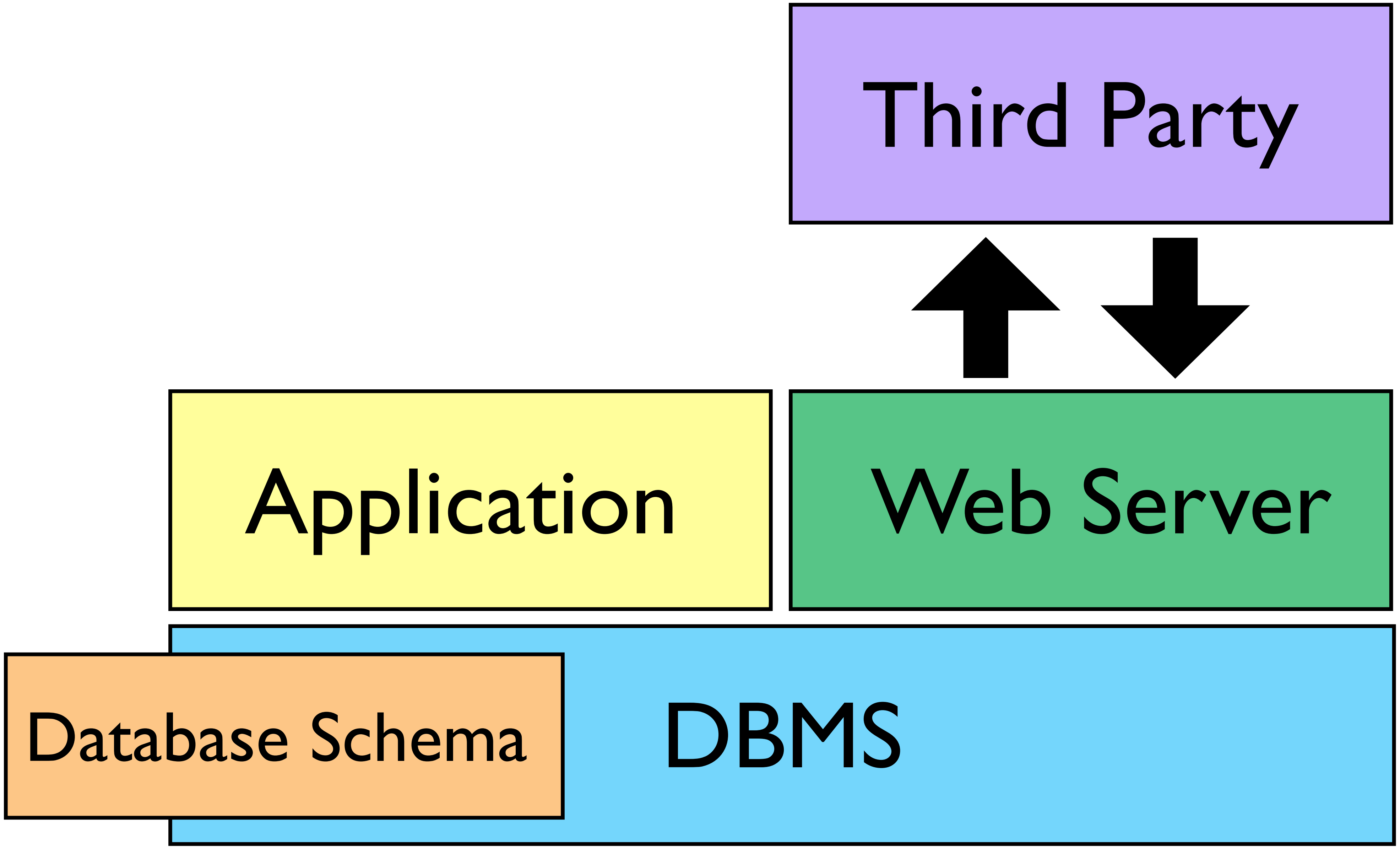
Why Test Database Structure?



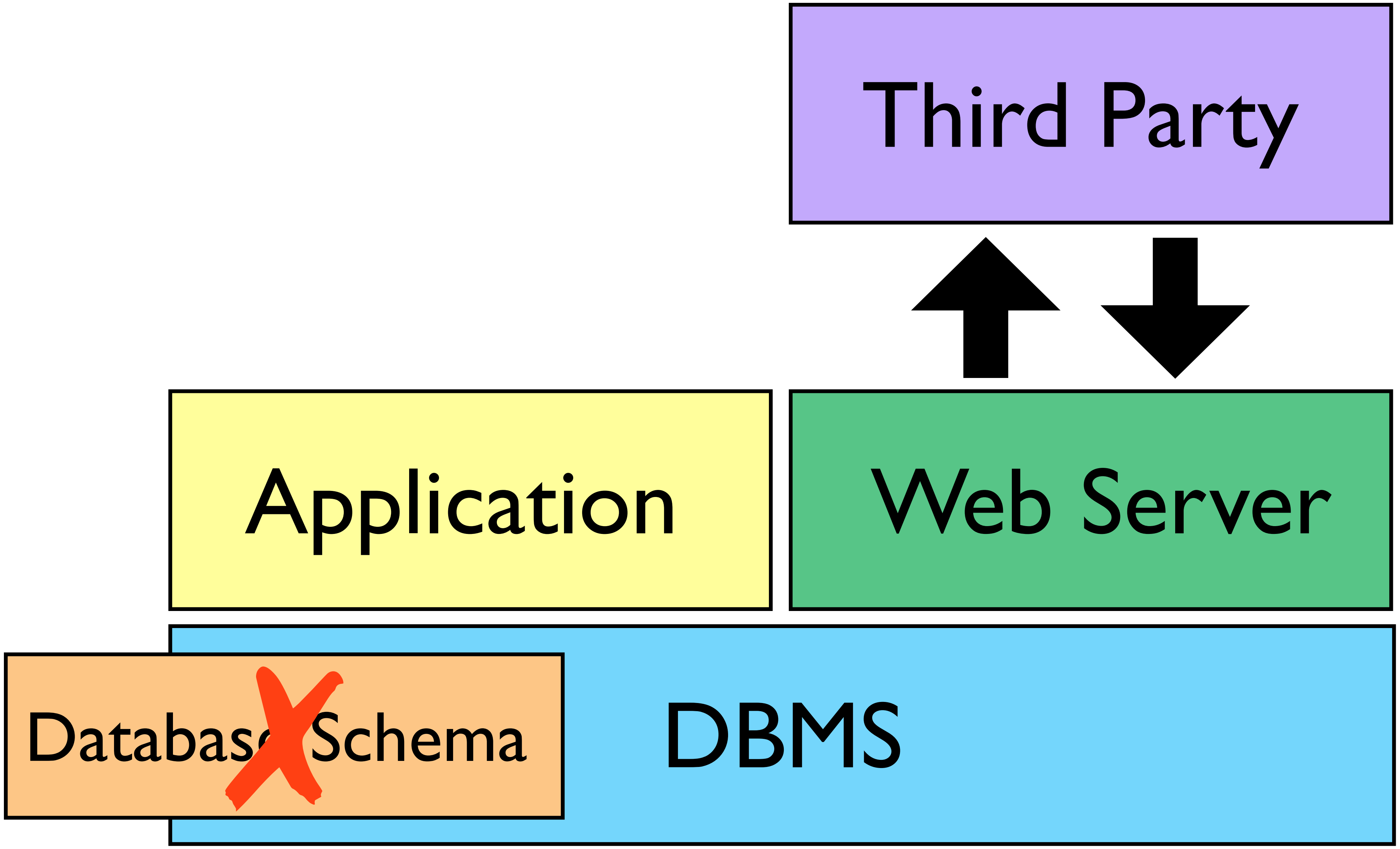
Why Test Database Structure?



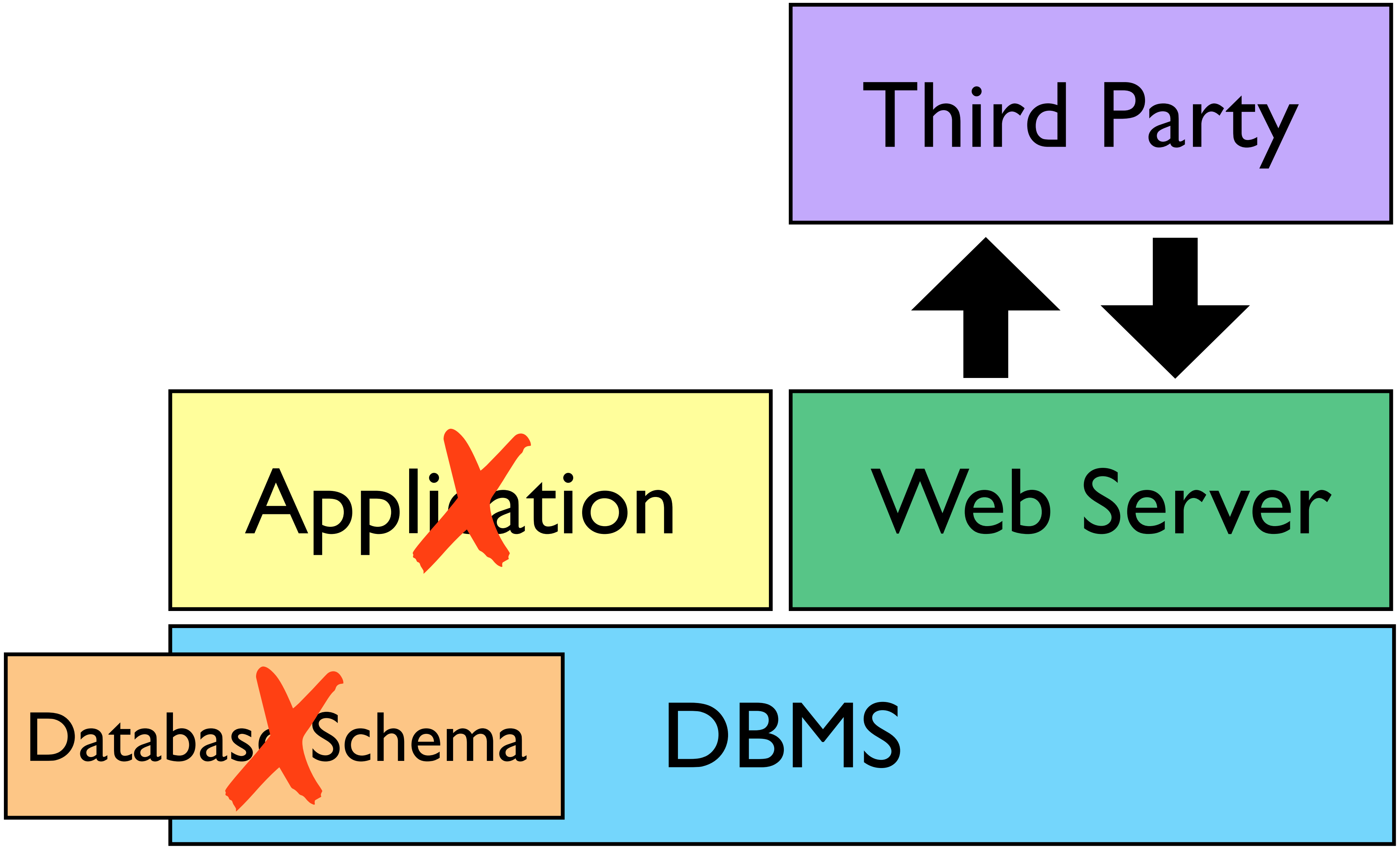
Why Test Database Structure?



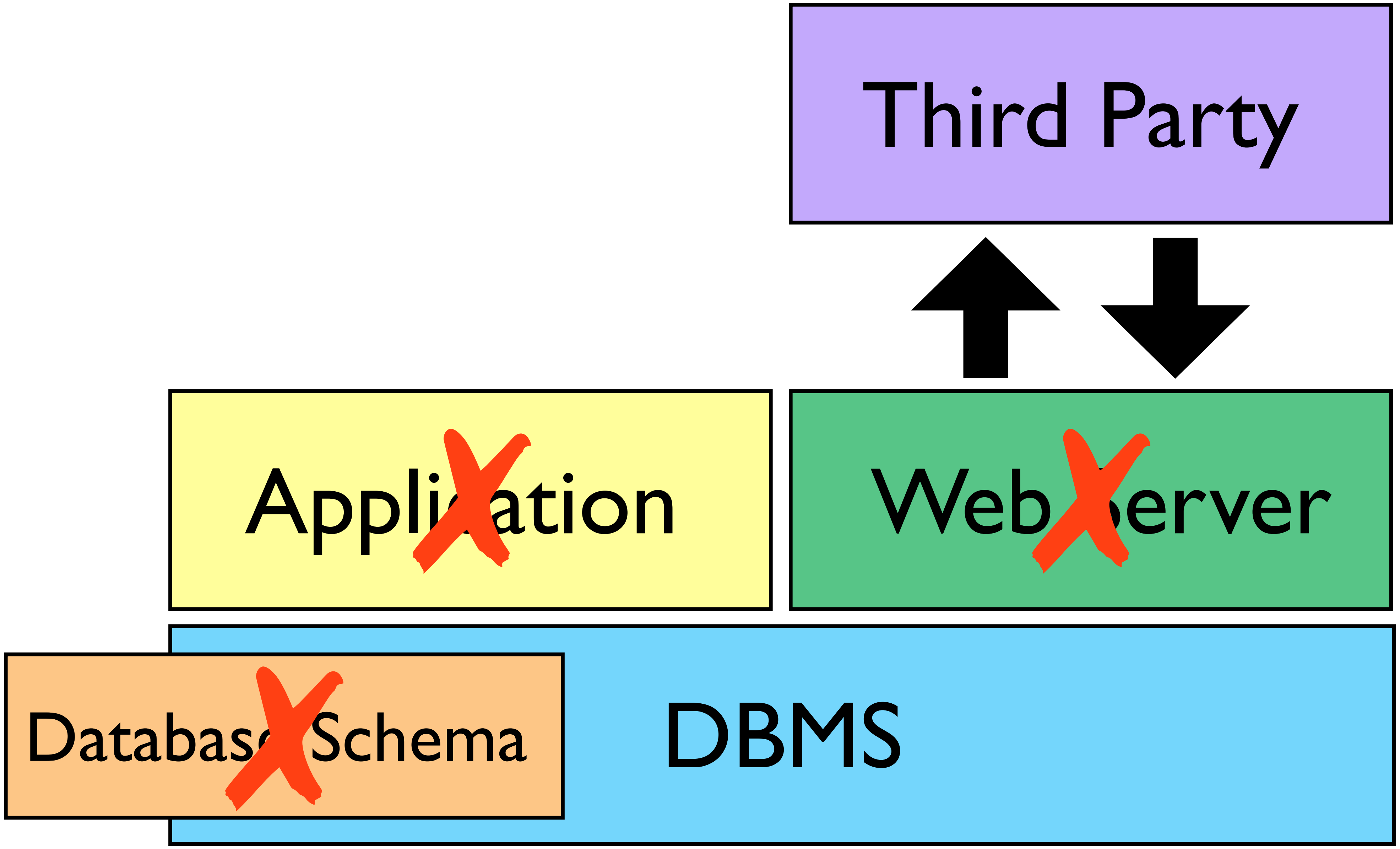
Why Test Database Structure?



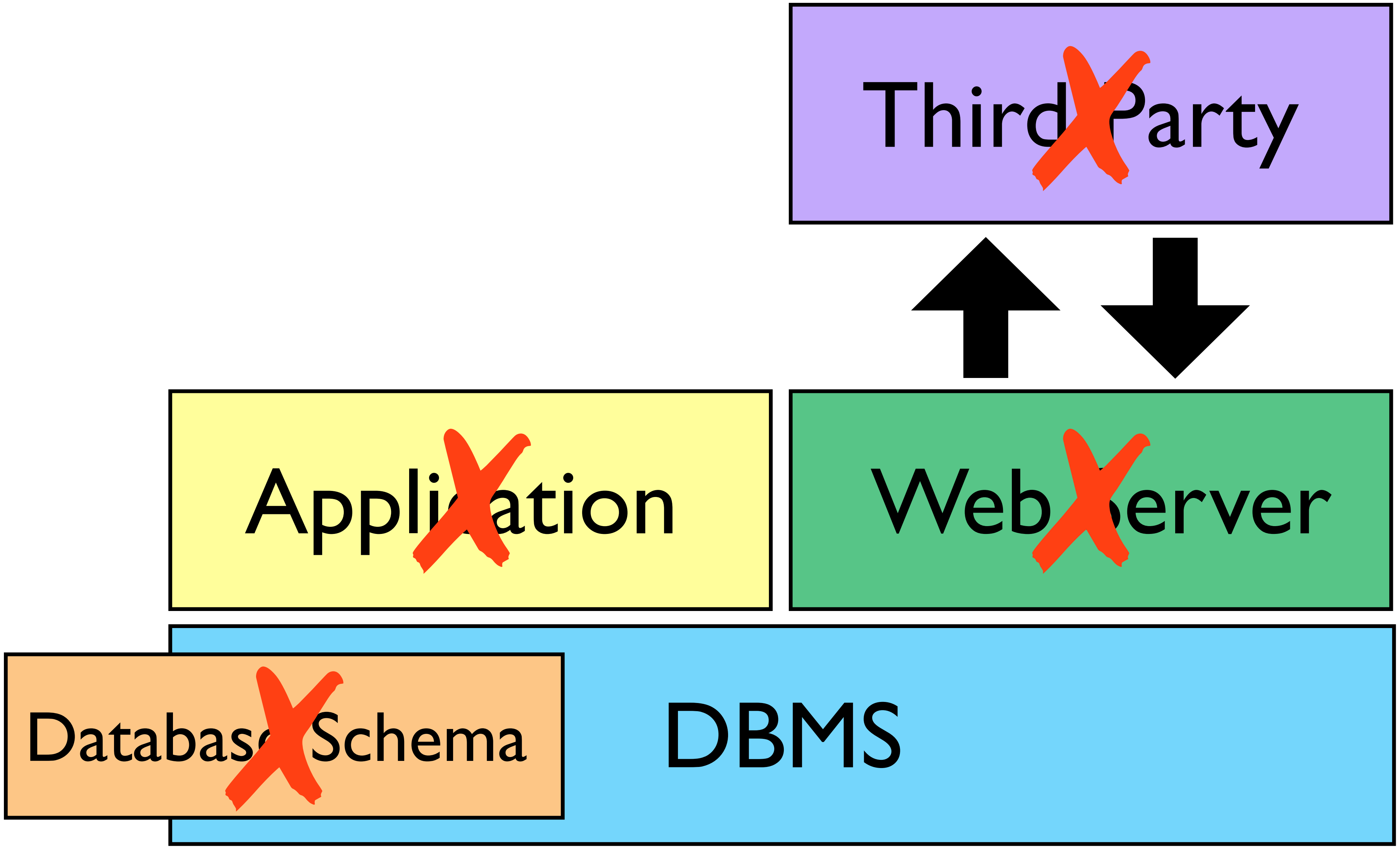
Why Test Database Structure?



Why Test Database Structure?



Why Test Database Structure?



Mutation Analysis

Mutation Analysis

Test Suite

Application

Mutation Analysis

Test Suite

Database

Mutation Analysis

Insert Statements

Database

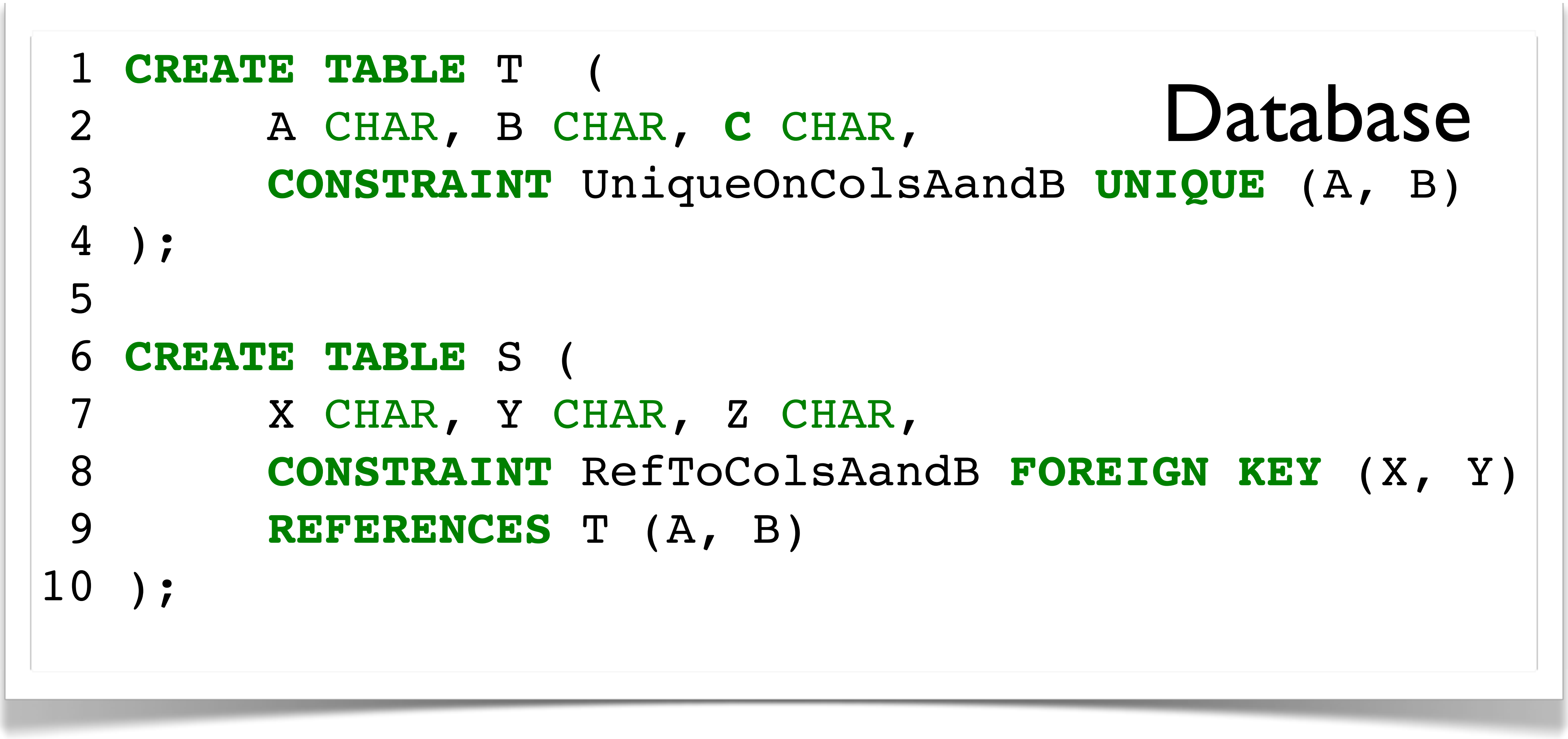
Mutation Analysis

Insert Statements

Database

Mutation Analysis

Insert Statements



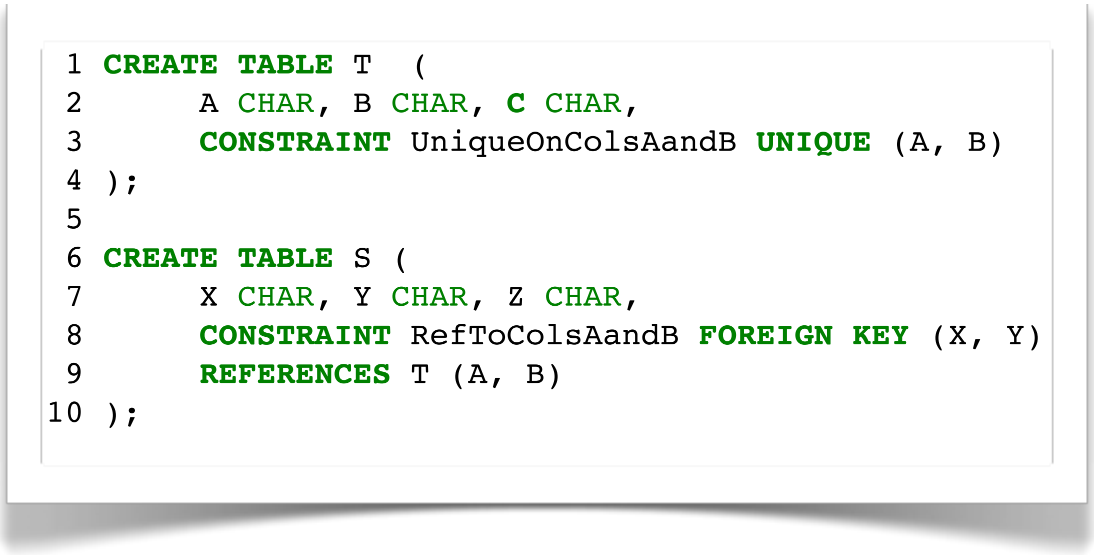
```sql
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```

Database

Mutation Analysis

Insert Statements

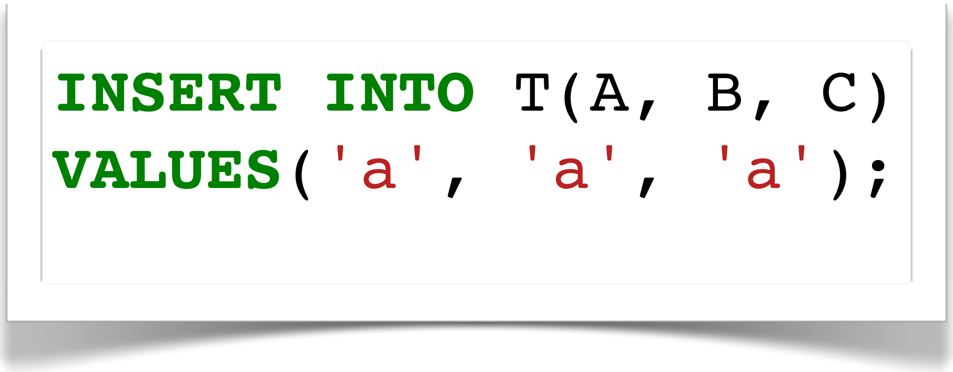
Database



```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```

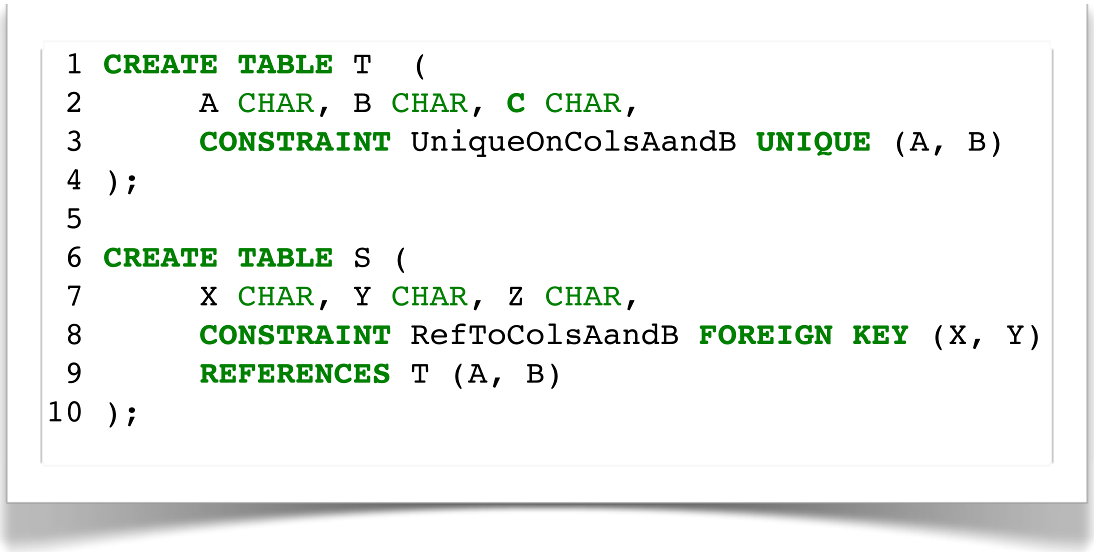
Mutation Analysis

Insert Statements



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

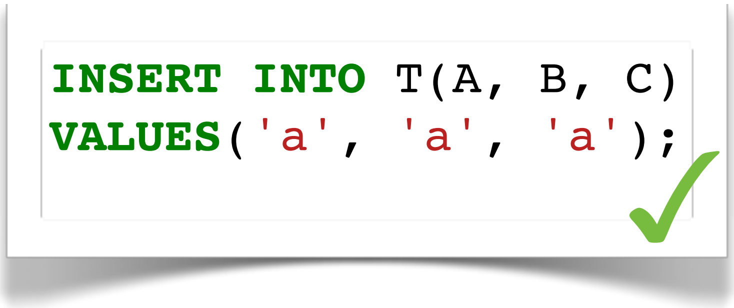
Database



```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```

Mutation Analysis

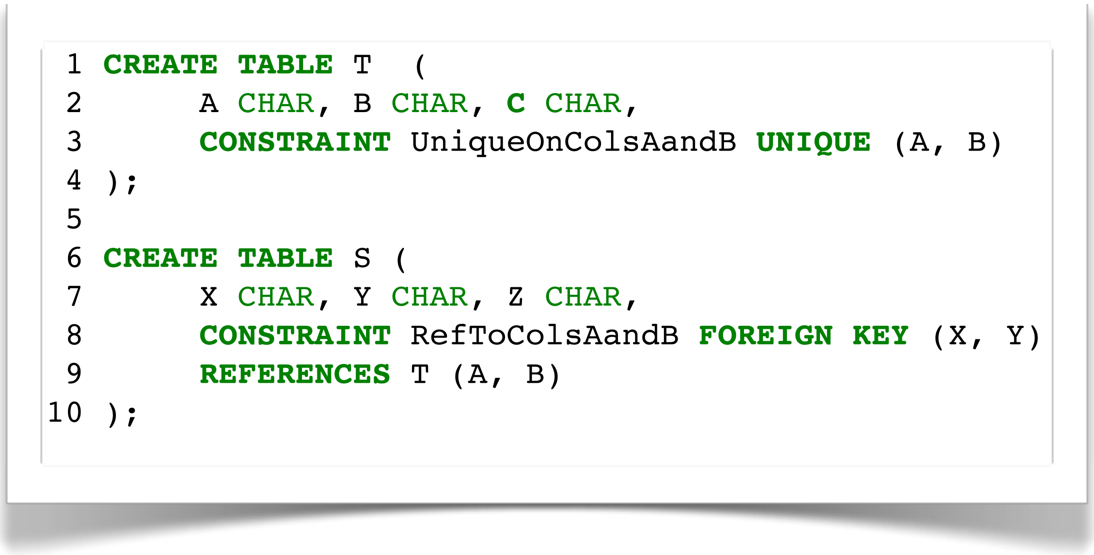
Insert Statements



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓

Database

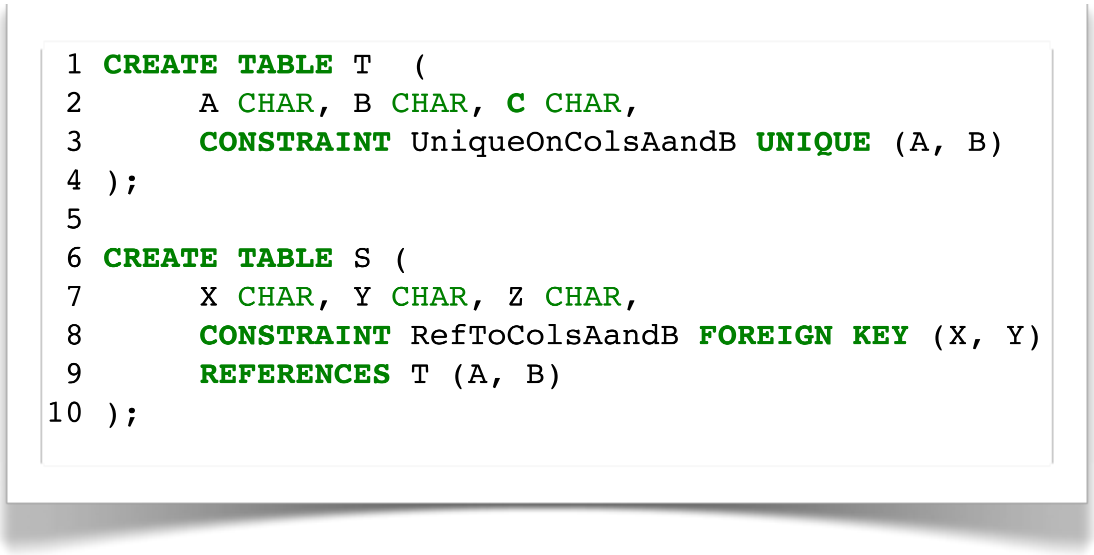


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```

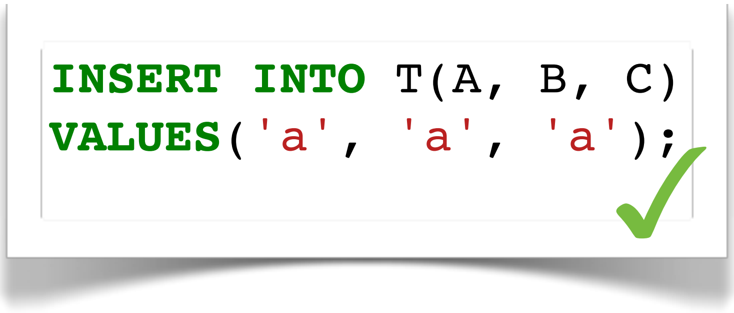
Mutation Analysis

Insert Statements

Database



```sql
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```

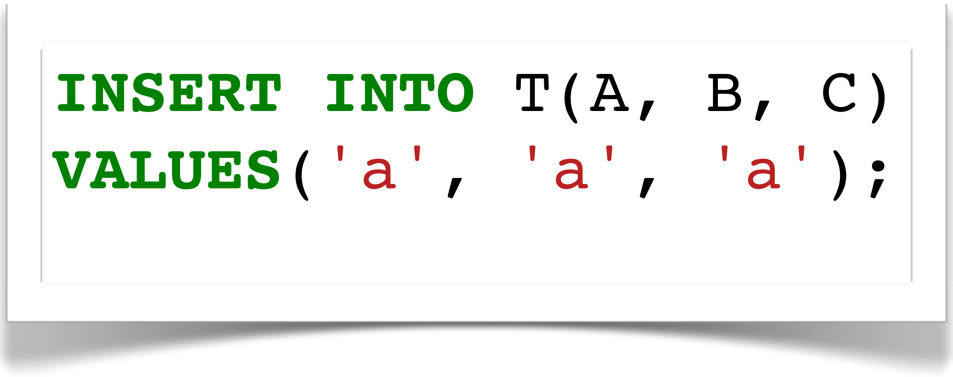


```sql
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓

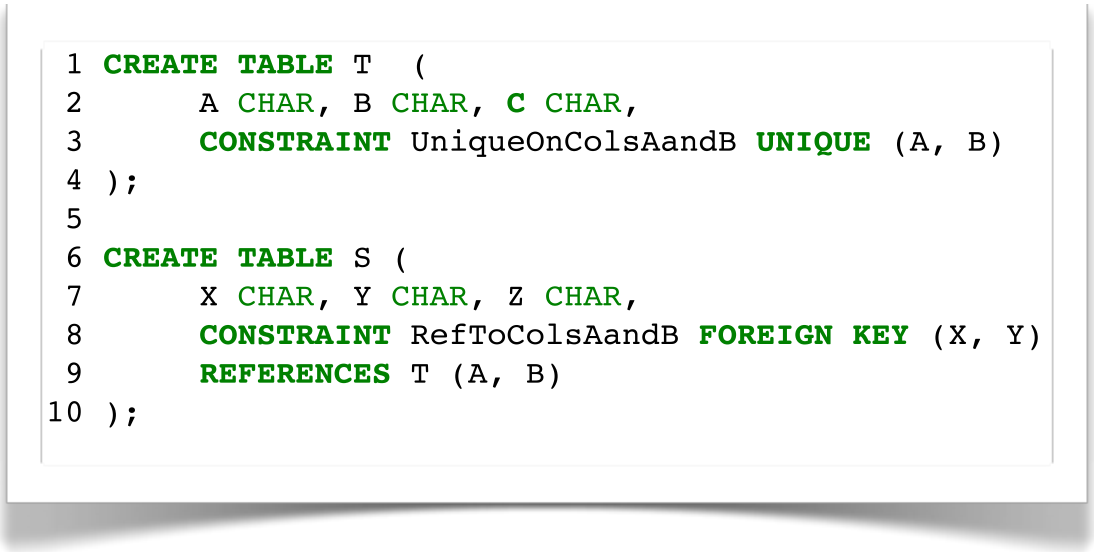
Mutation Analysis

Insert Statements

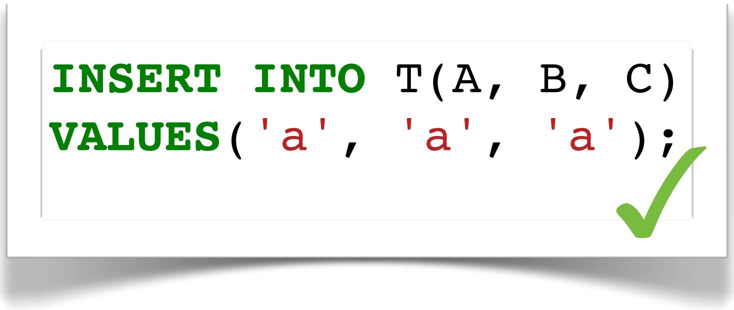


```sql
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

Database



```sql
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```



```sql
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

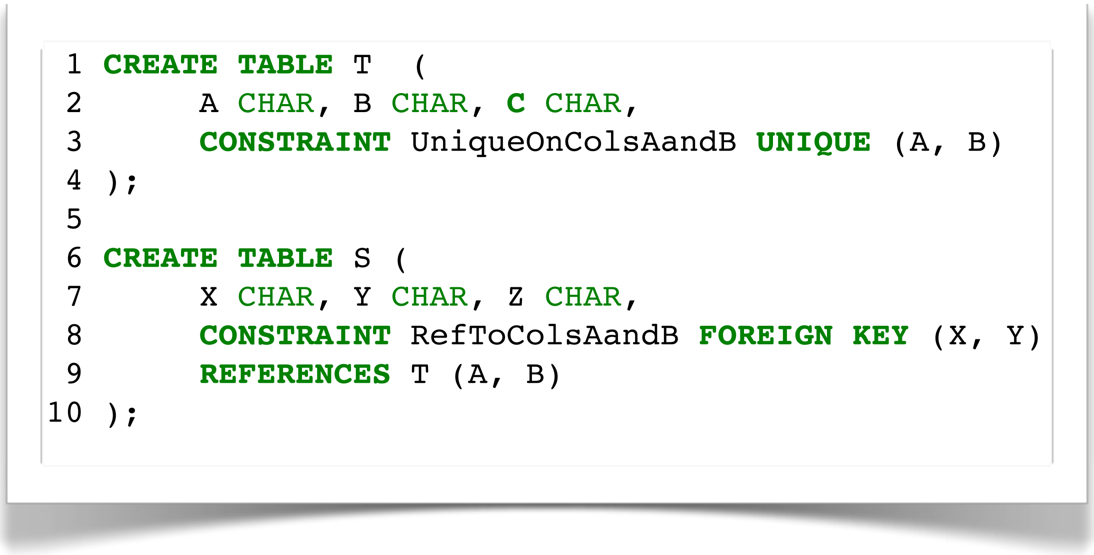
✓

Chris J. Wright - c.wright@dcs.shef.ac.uk

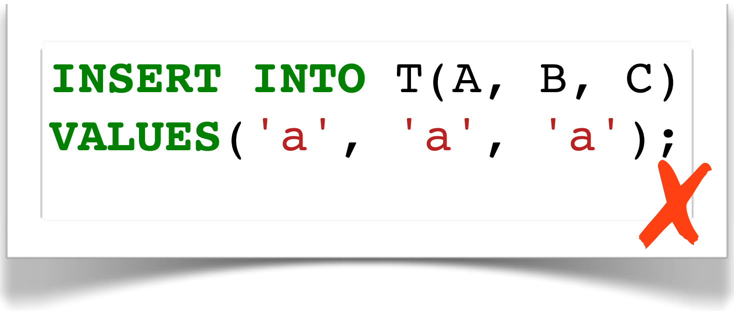
Mutation Analysis

Insert Statements

Database

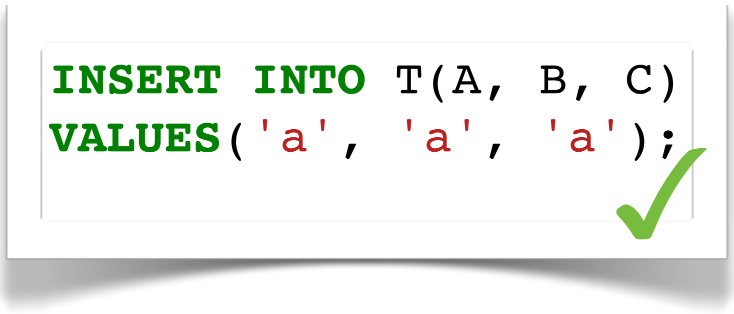


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✗



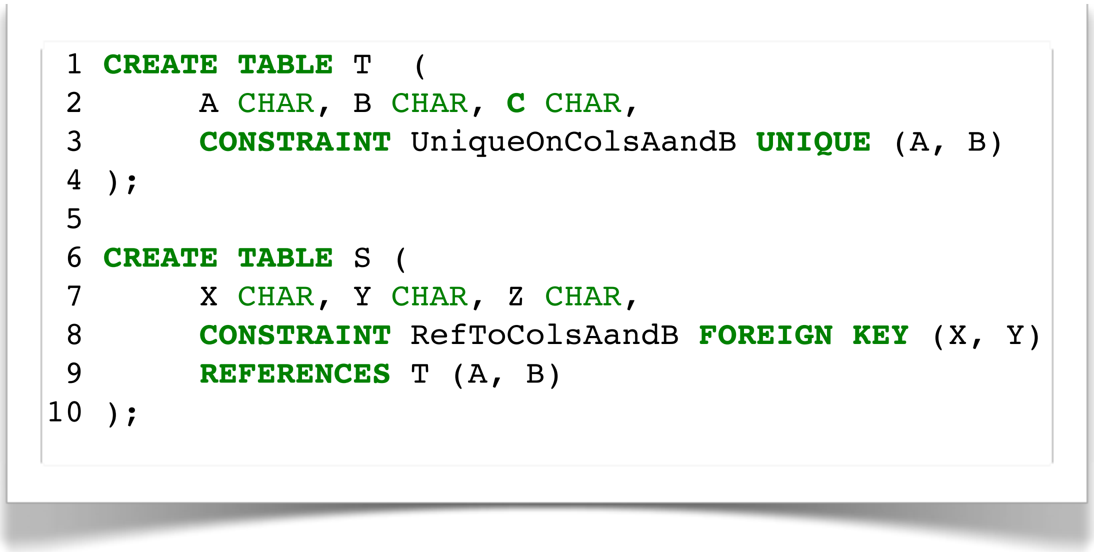
```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓

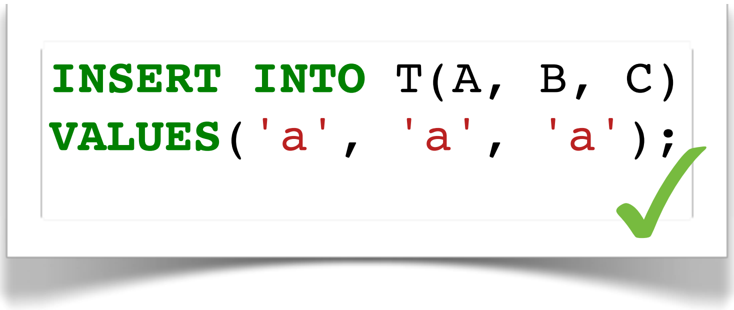
Mutation Analysis

Insert Statements

Database

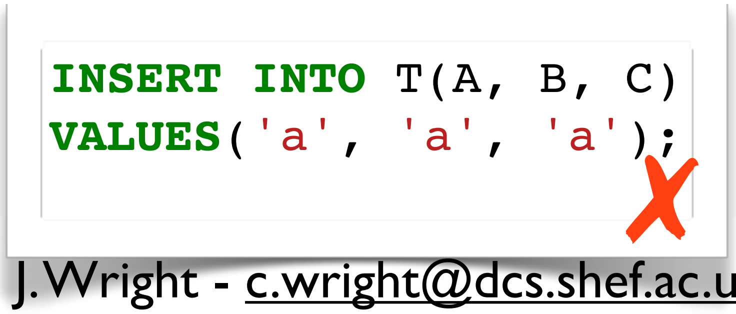


```sql
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```



```sql
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓

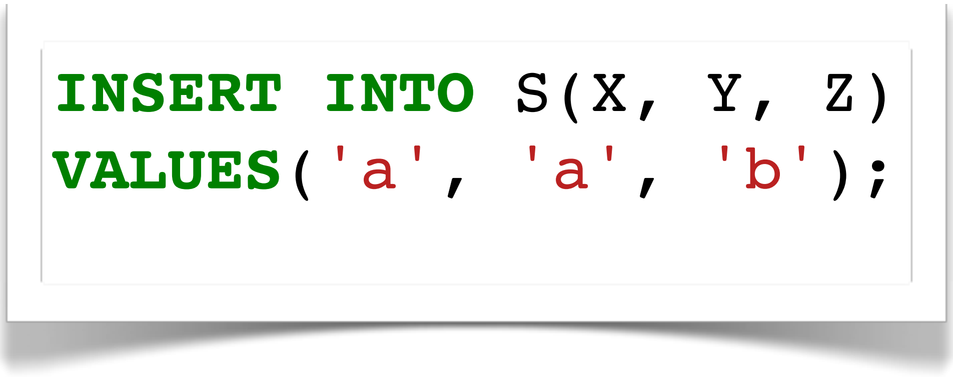


```sql
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✗

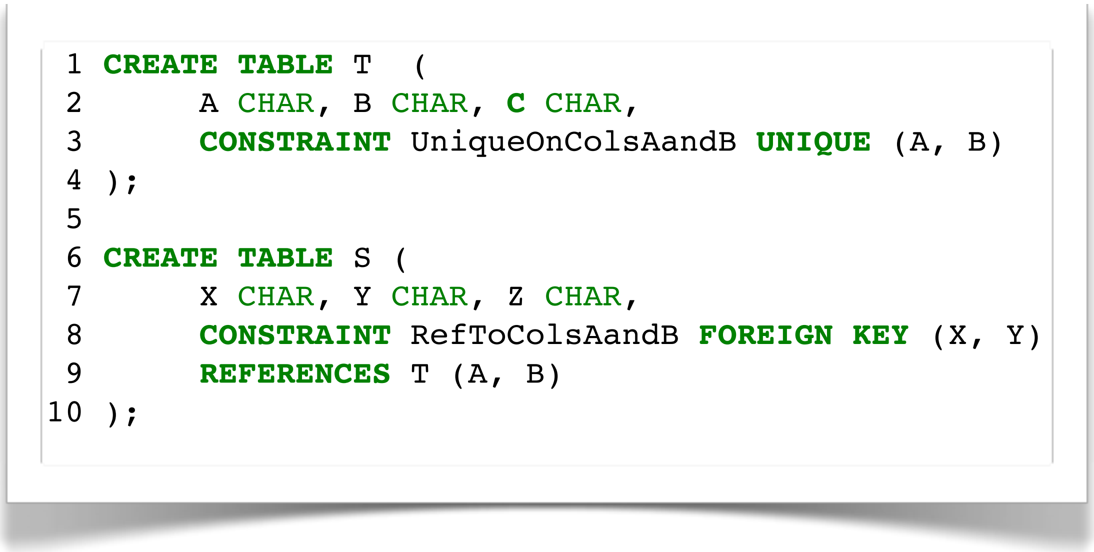
Mutation Analysis

Insert Statements

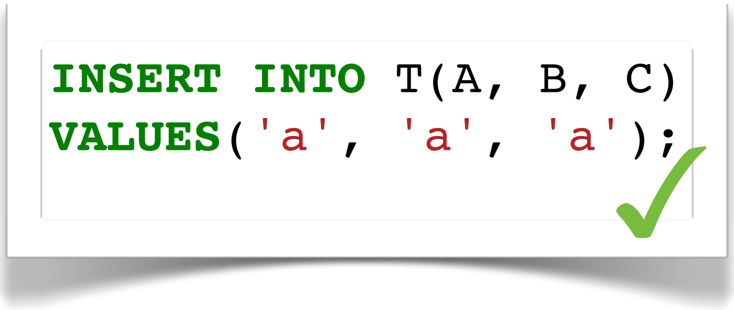


```
INSERT INTO S(X, Y, Z)
VALUES('a', 'a', 'b');
```

Database

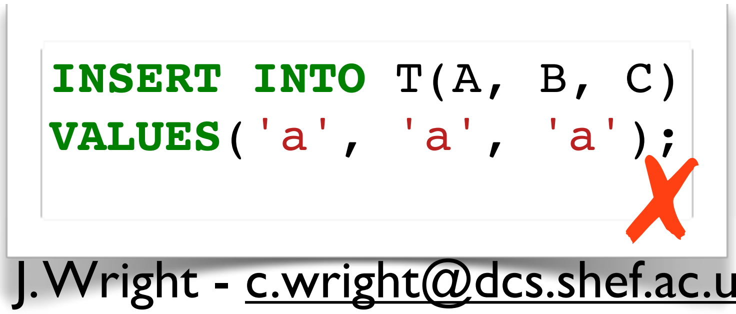


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓



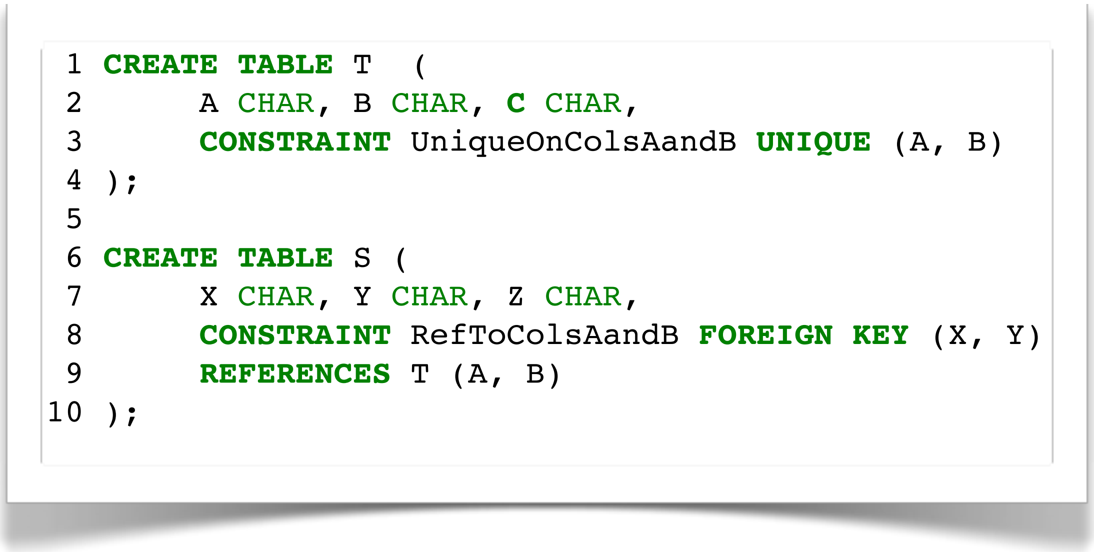
```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✗

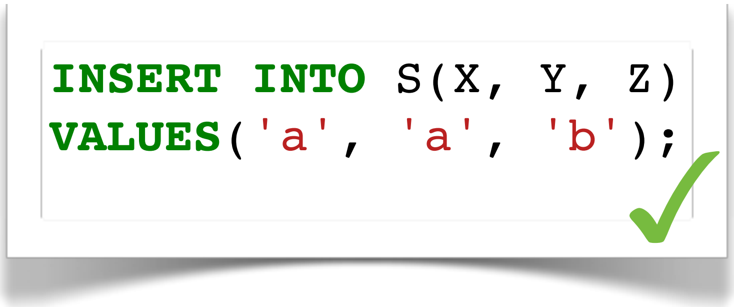
Mutation Analysis

Insert Statements

Database

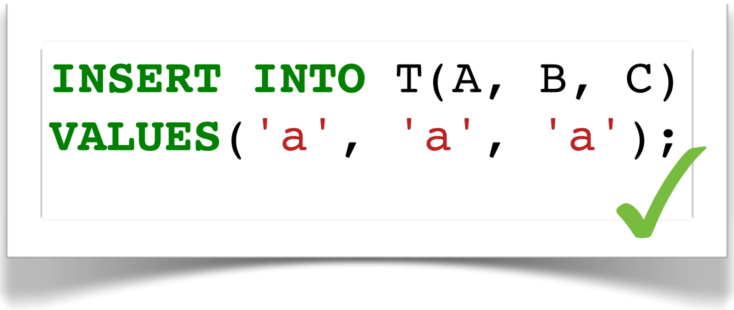


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```



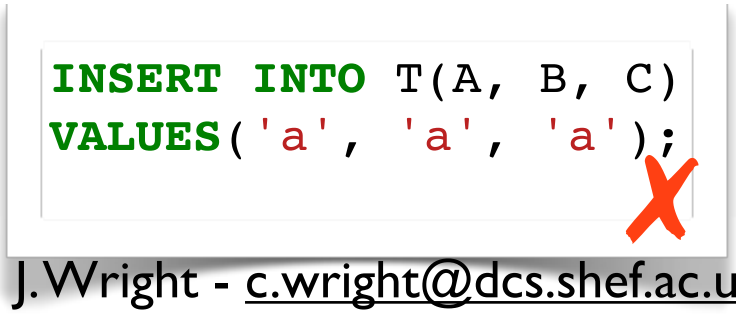
```
INSERT INTO S(X, Y, Z)
VALUES('a', 'a', 'b');
```

✓



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓



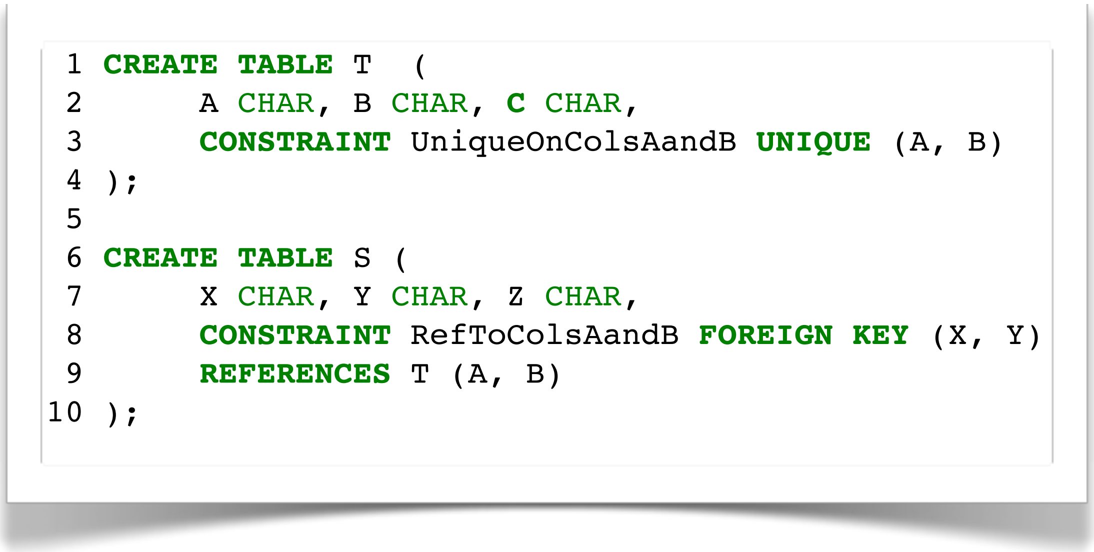
```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✗

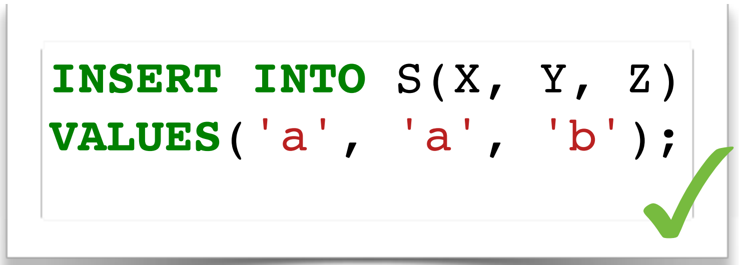
Mutation Analysis

Insert Statements

Database

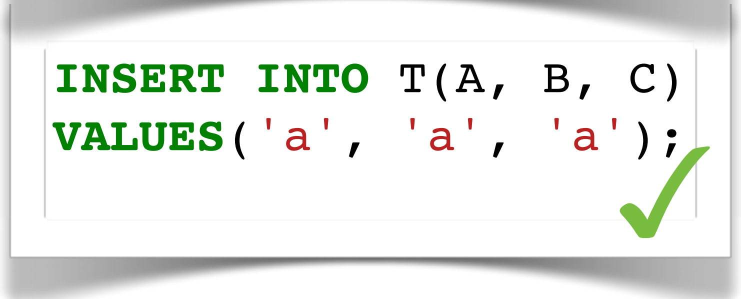


```sql
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```



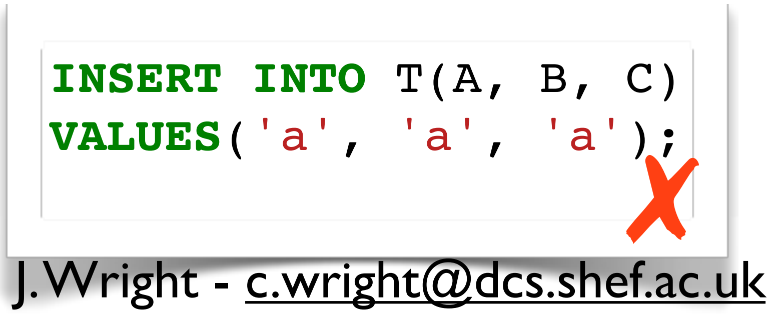
```sql
INSERT INTO S(X, Y, Z)
VALUES('a', 'a', 'b');
```

✓



```sql
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓

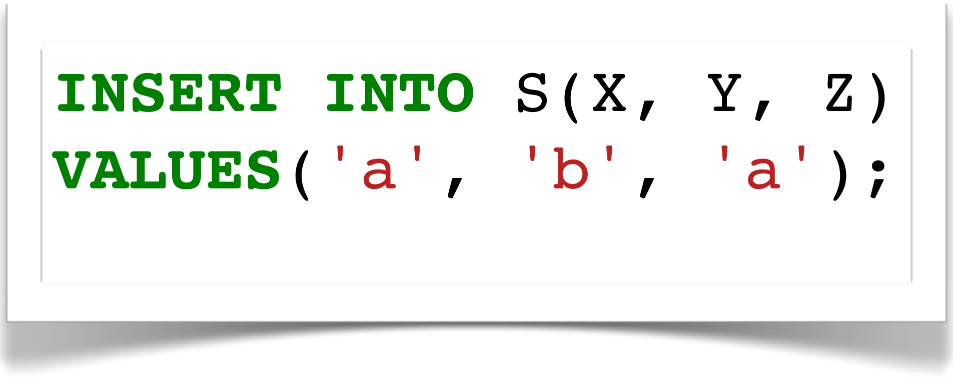


```sql
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✗

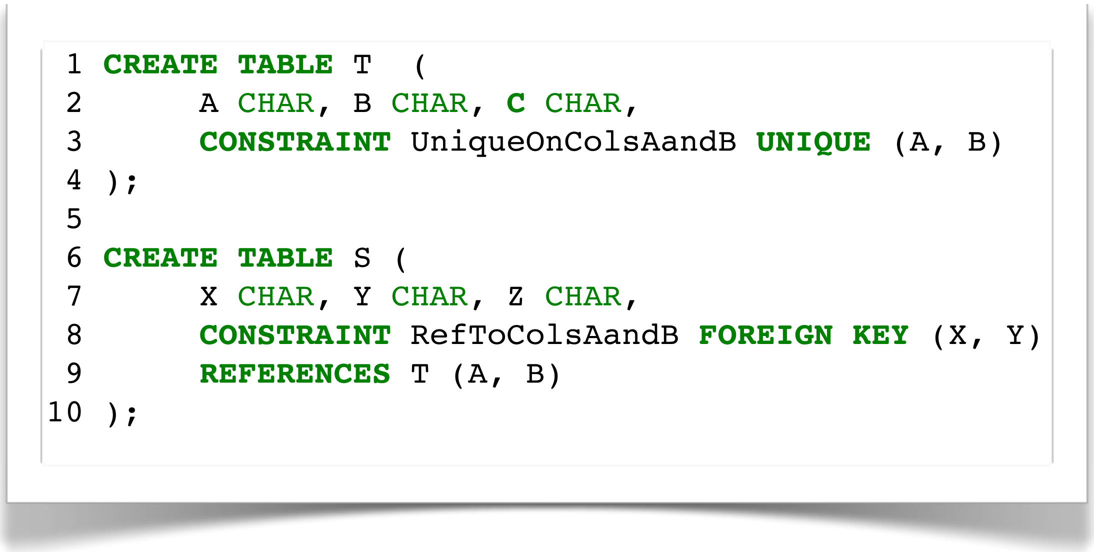
Mutation Analysis

Insert Statements

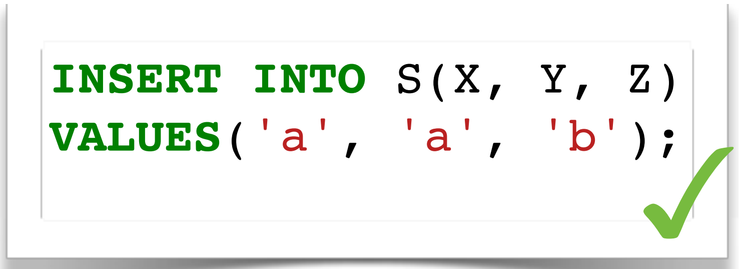


```
INSERT INTO S(X, Y, Z)
VALUES('a', 'b', 'a');
```

Database

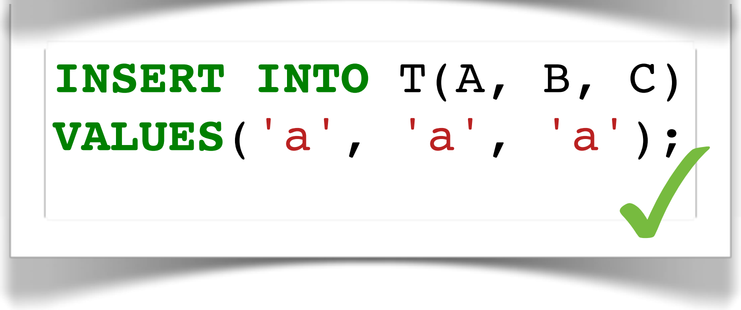


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```



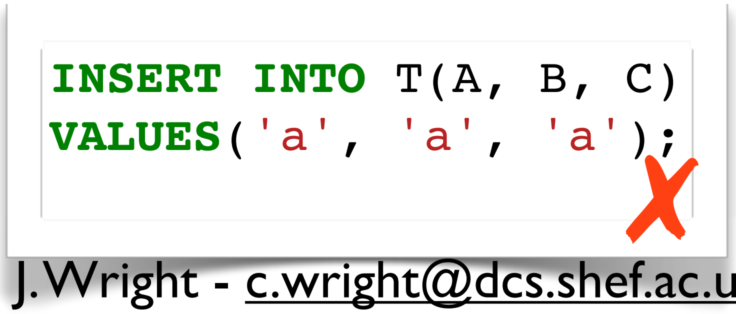
```
INSERT INTO S(X, Y, Z)
VALUES('a', 'a', 'b');
```

✓



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓



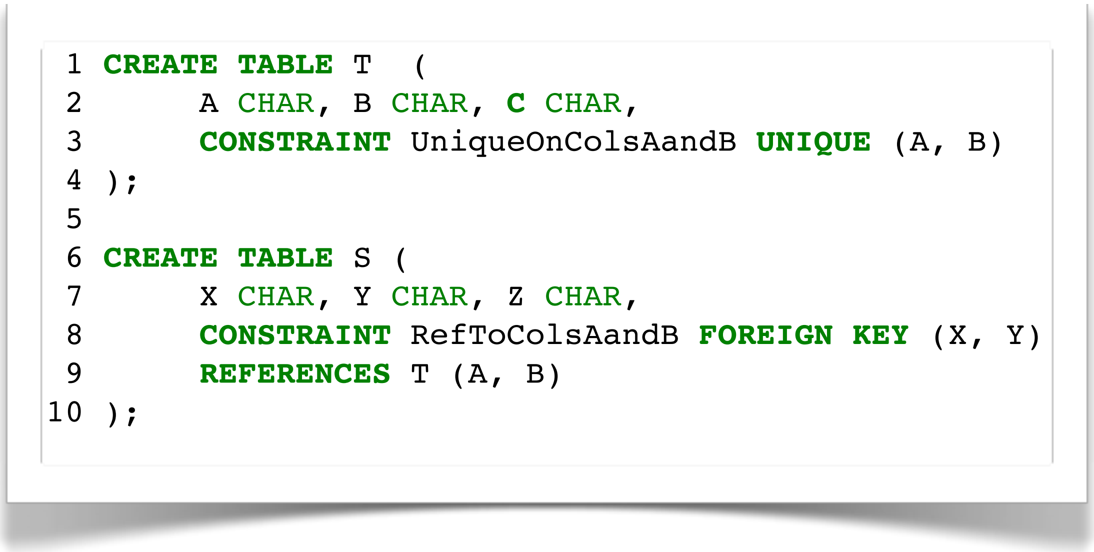
```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✗

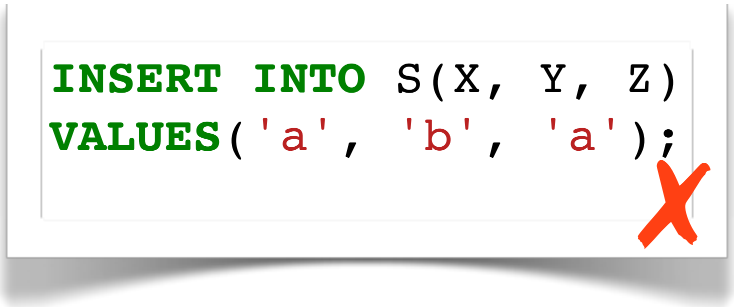
Mutation Analysis

Insert Statements

Database

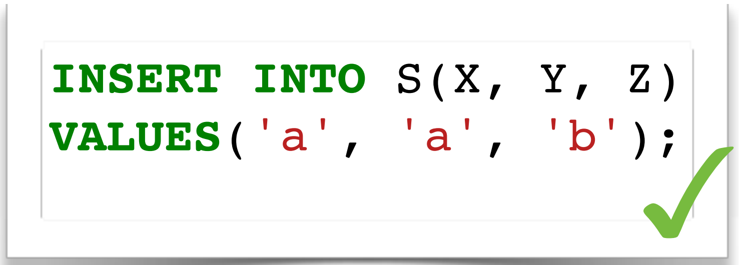


```
CREATE TABLE T  (
      A CHAR, B CHAR, C CHAR,
      CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
      X CHAR, Y CHAR, Z CHAR,
      CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
      REFERENCES T (A, B)
);
```



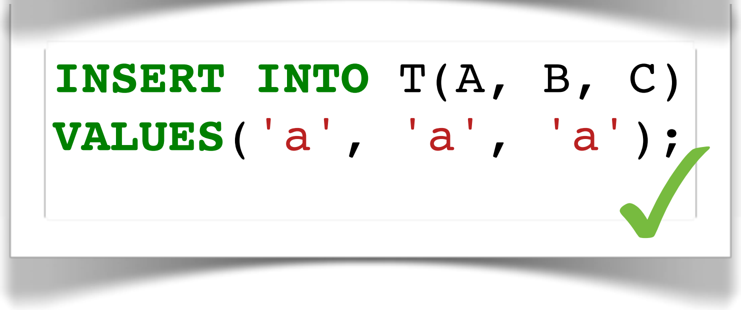
```
INSERT INTO S(X, Y, Z)
VALUES('a', 'b', 'a');
```

✗



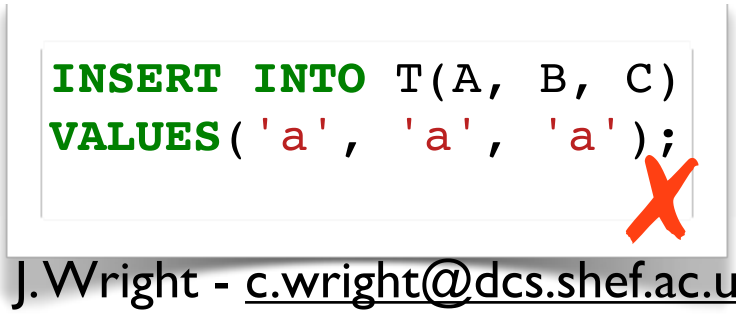
```
INSERT INTO S(X, Y, Z)
VALUES('a', 'a', 'b');
```

✓



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓



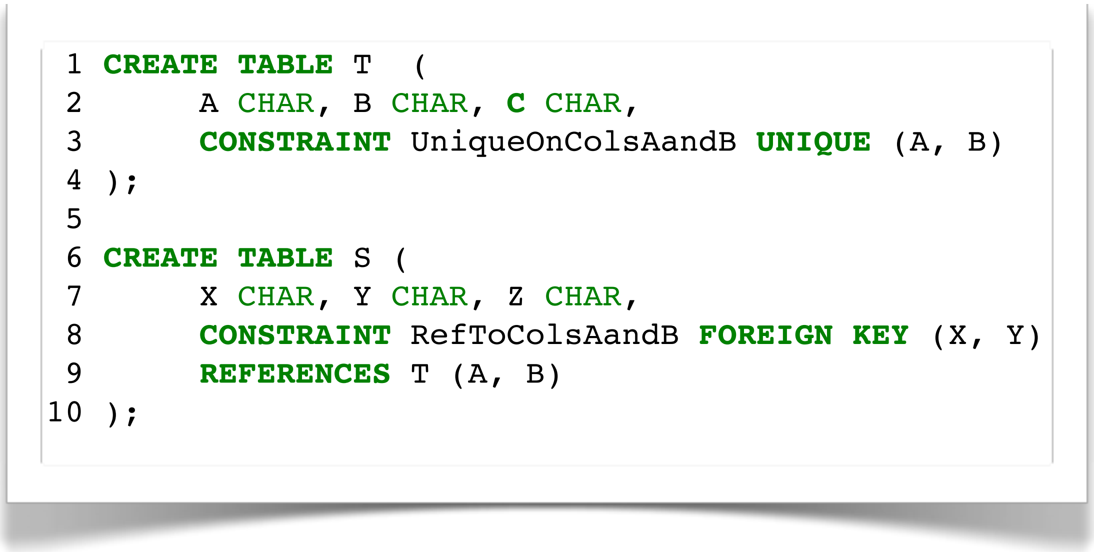
```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✗

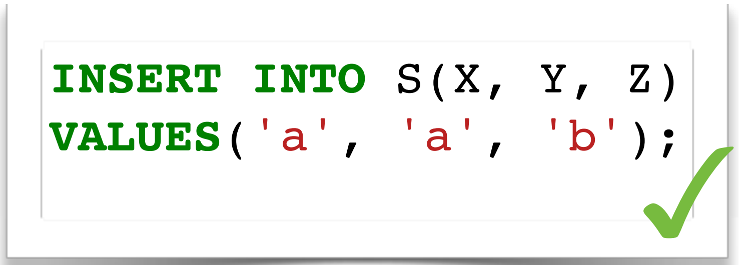
Mutation Analysis

Insert Statements

Database

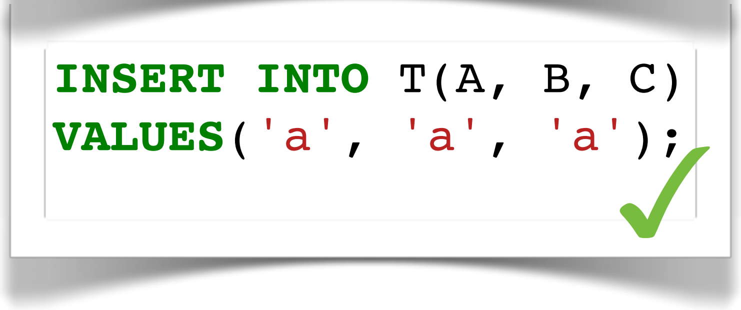


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```



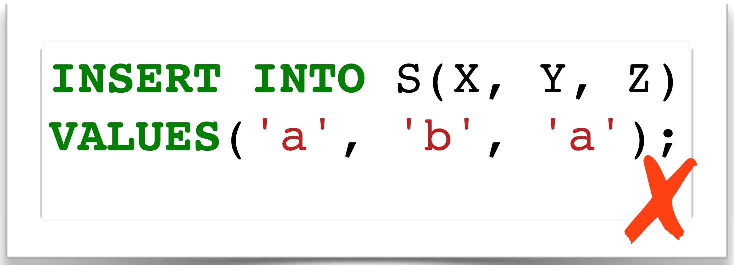
```
INSERT INTO S(X, Y, Z)
VALUES('a', 'a', 'b');
```

✓



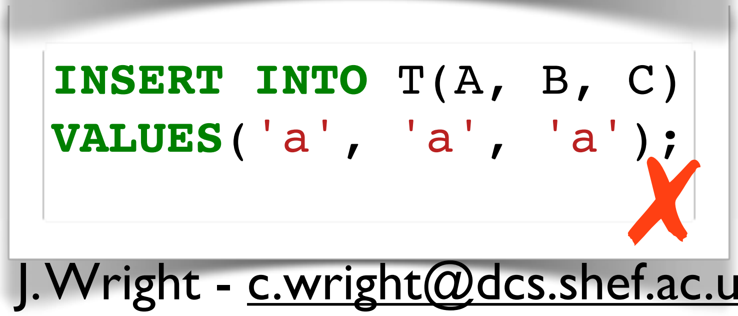
```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓



```
INSERT INTO S(X, Y, Z)
VALUES('a', 'b', 'a');
```

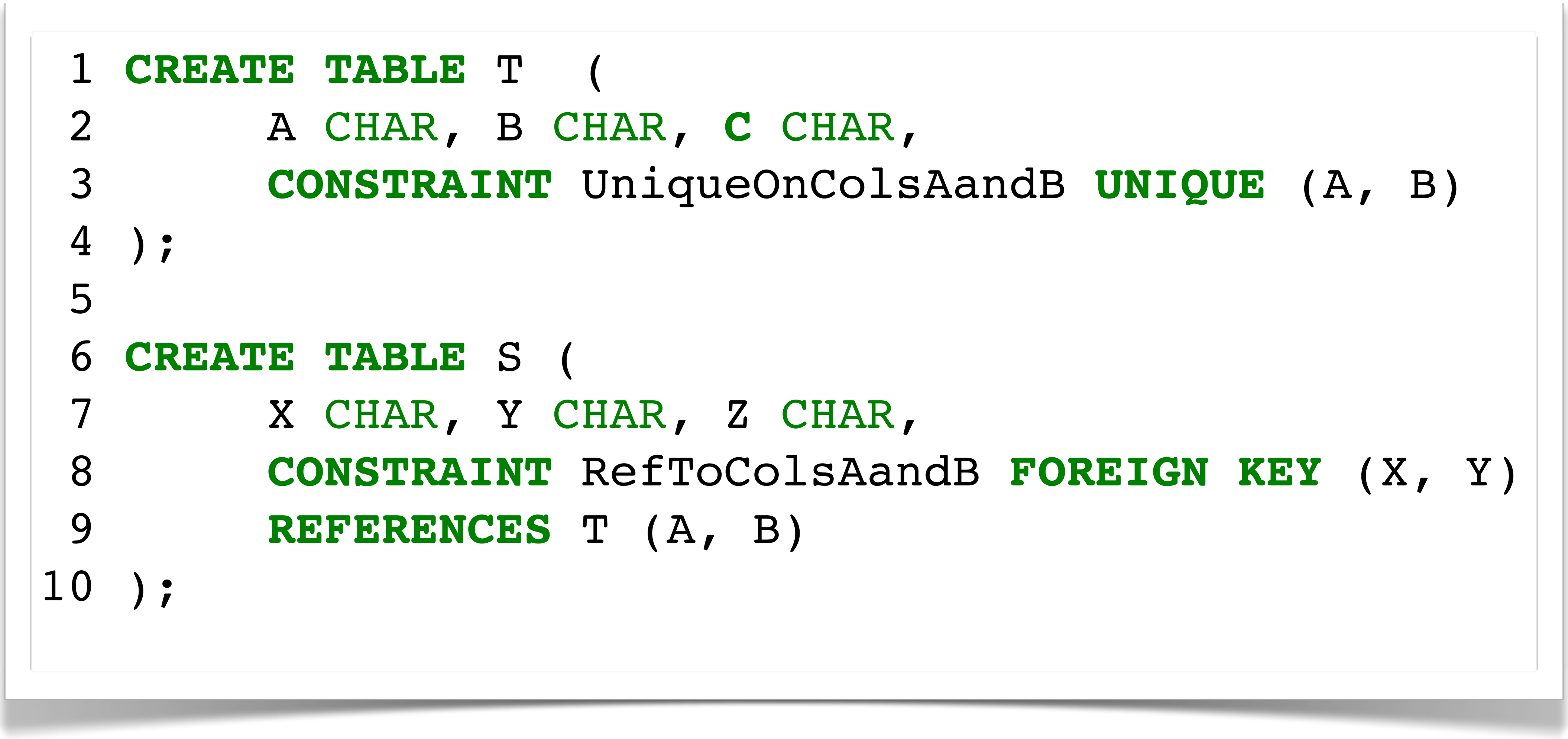
✗



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

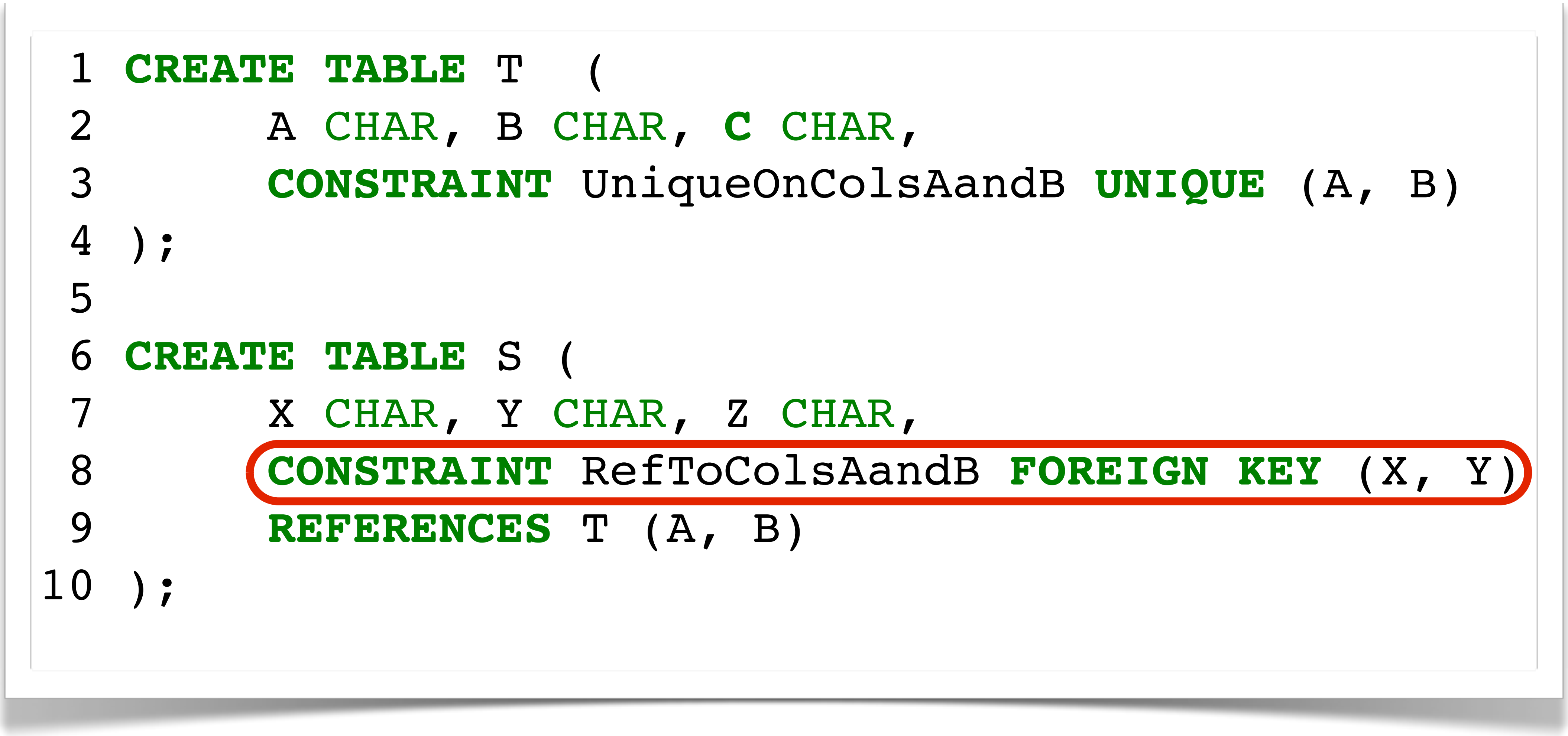
✗

Mutating the Structure



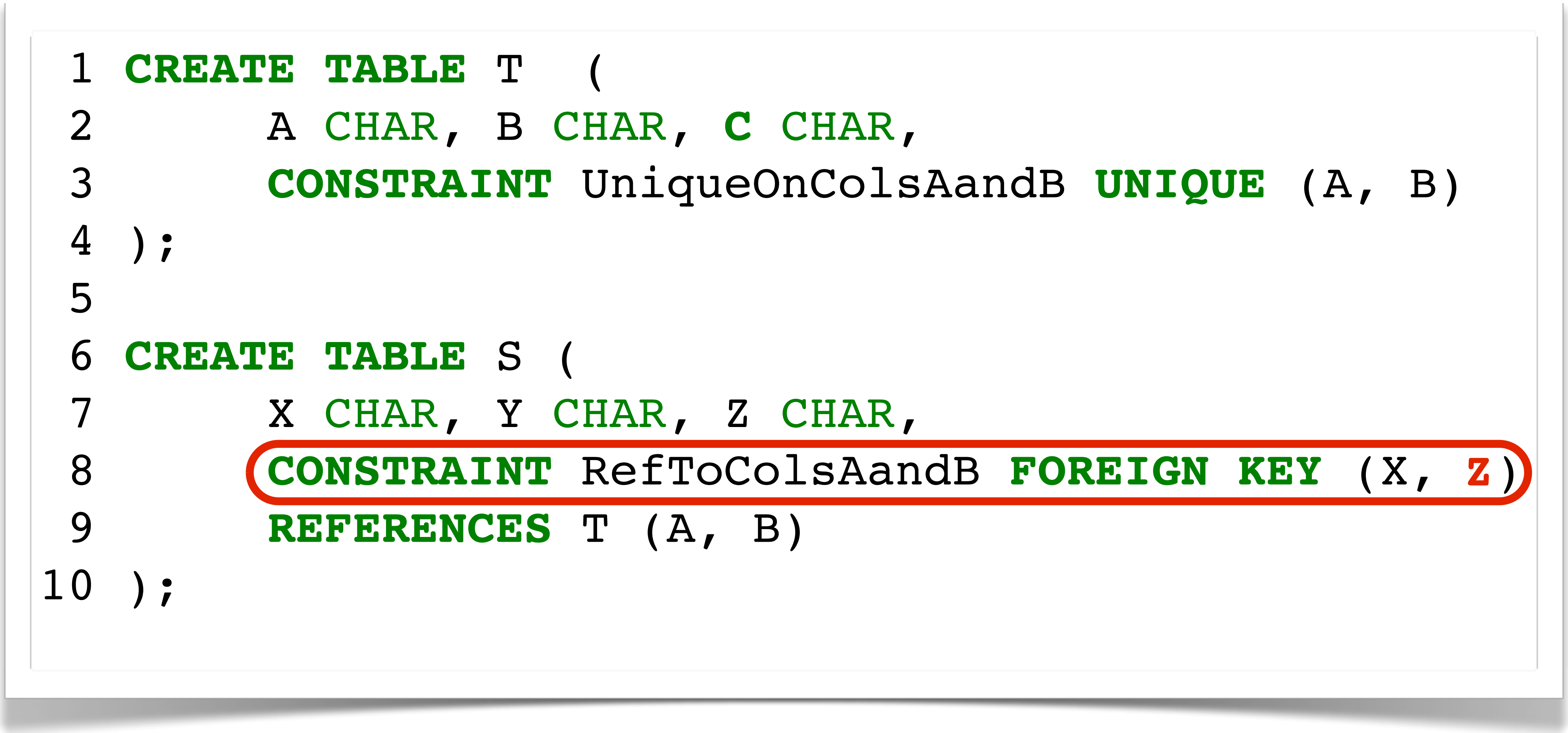
```sql
CREATE TABLE T  (
      A CHAR, B CHAR, C CHAR,
      CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
      X CHAR, Y CHAR, Z CHAR,
      CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
      REFERENCES T (A, B)
);
```

Mutating the Structure



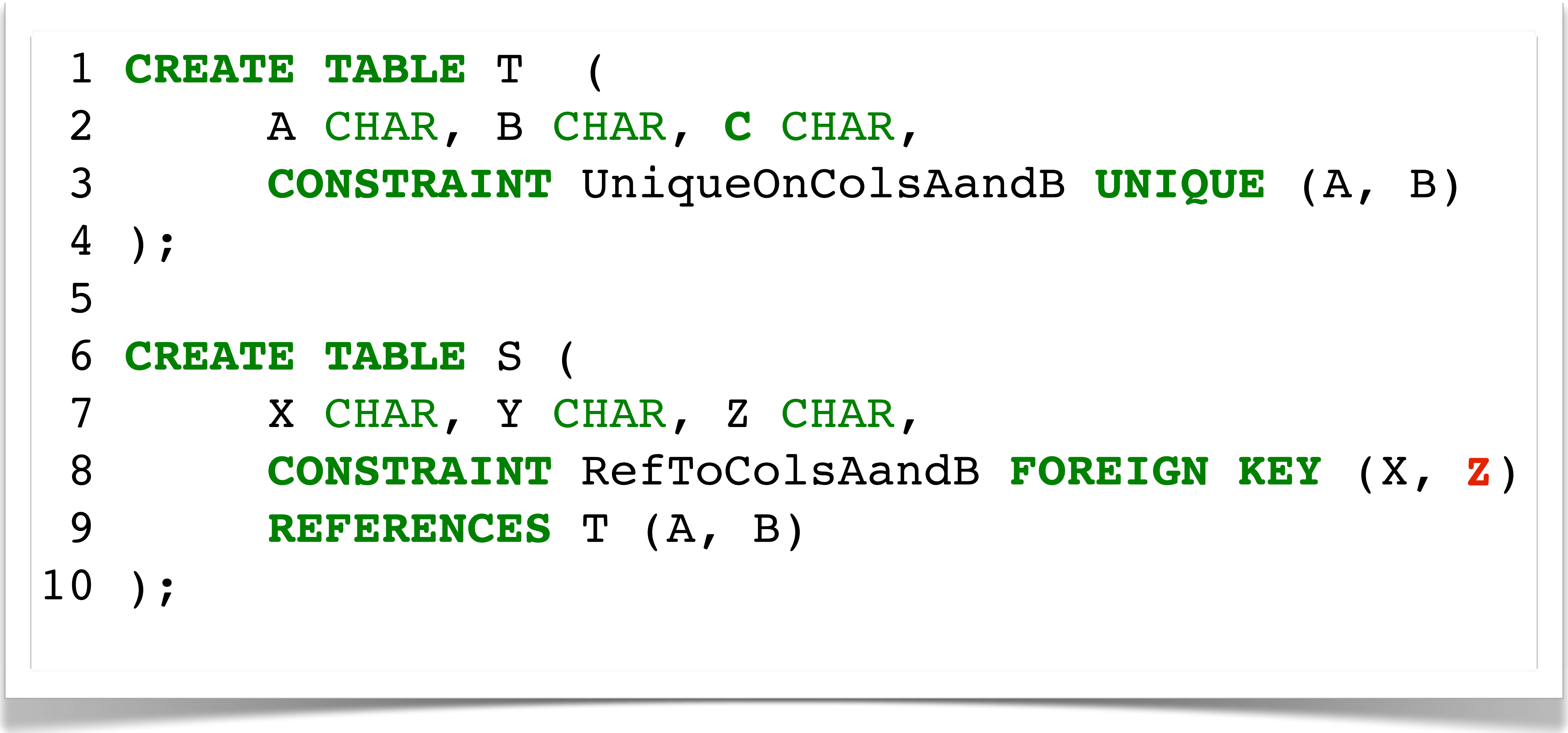
```sql
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```

Mutating the Structure



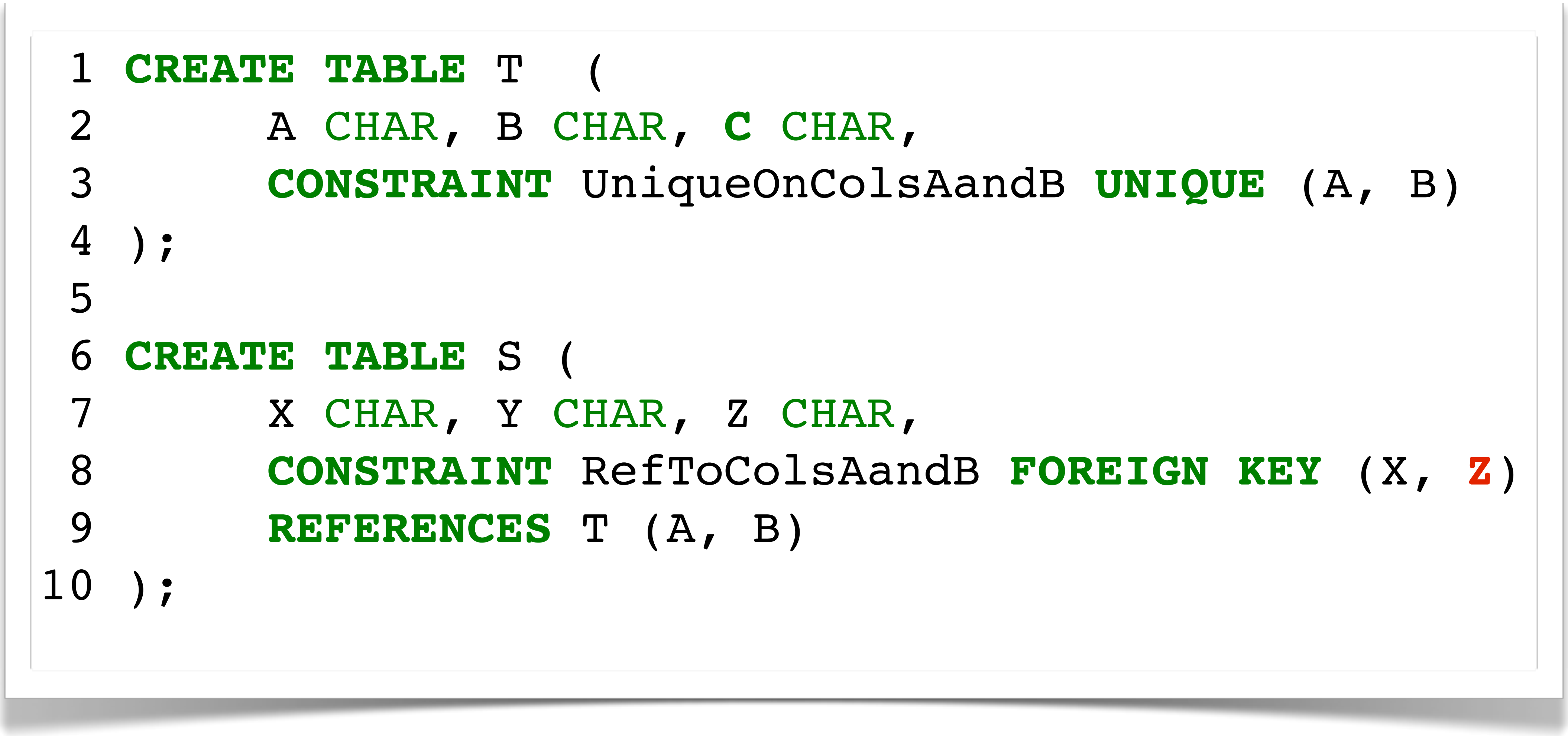
```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```

Mutating the Structure



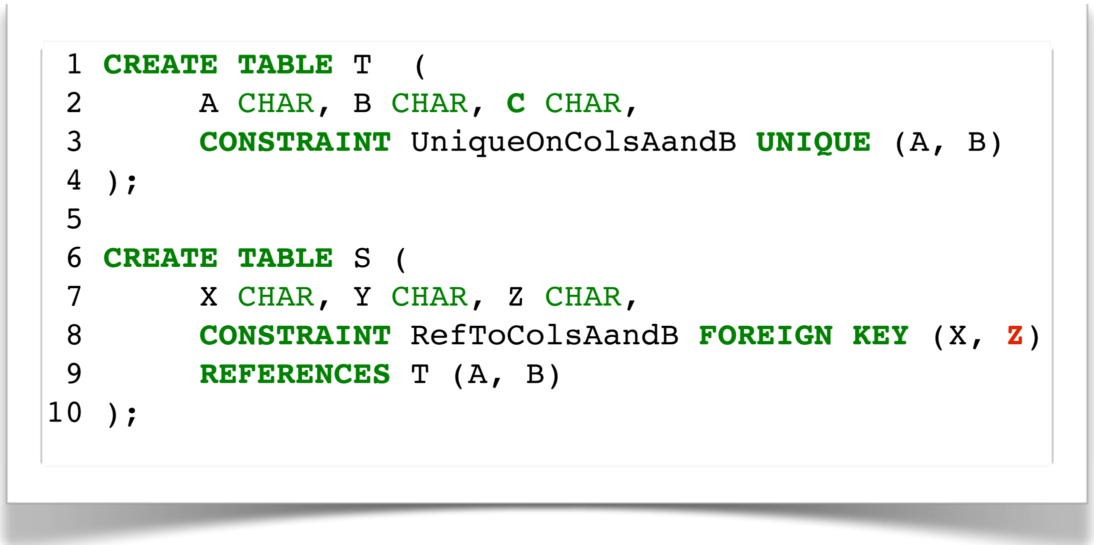
```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```

Mutation Analysis



```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```

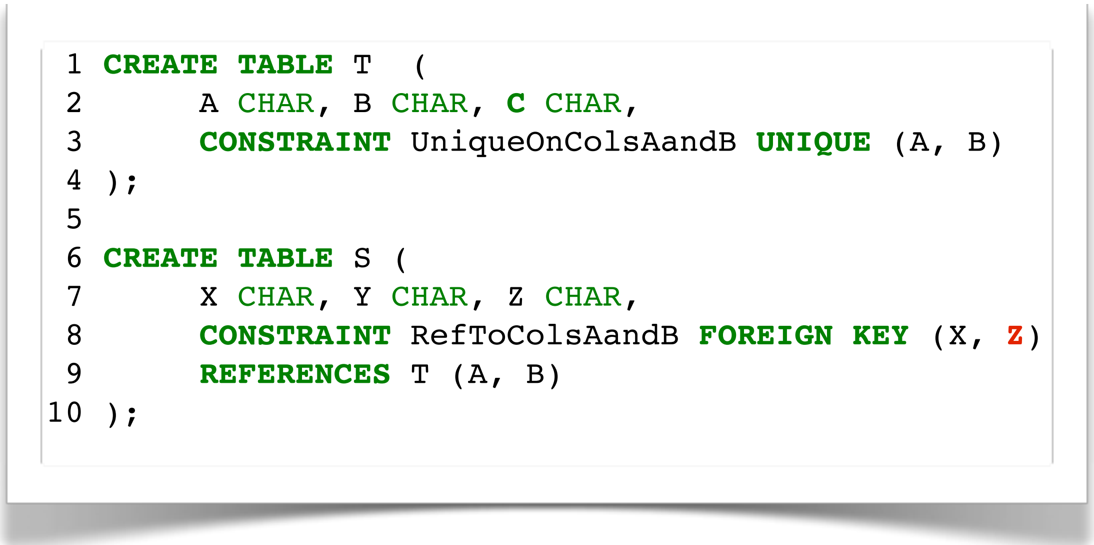
Mutation Analysis



```sql
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```

Mutation Analysis

Database (mutated)

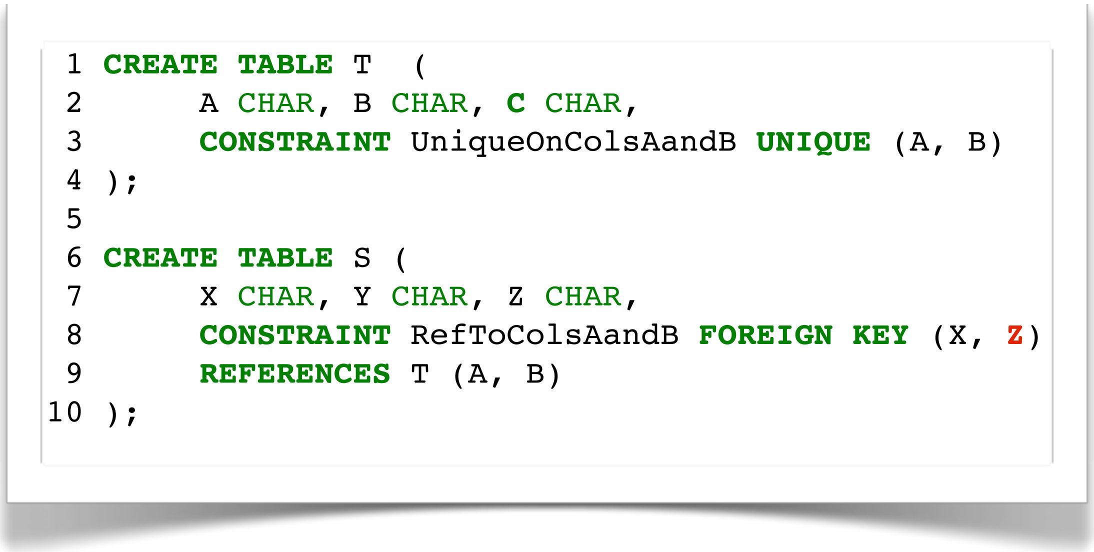


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```

Mutation Analysis

Insert Statements

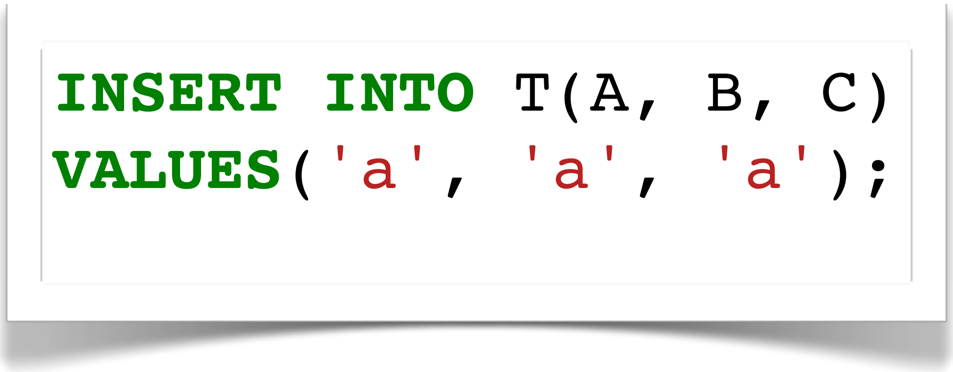
Database (mutated)



```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```

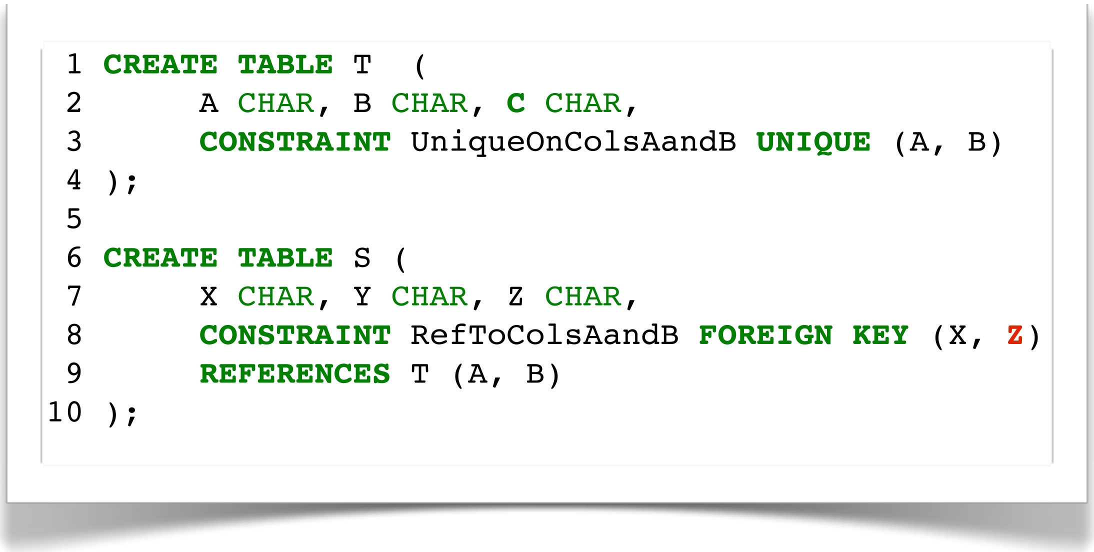
Mutation Analysis

Insert Statements



```sql
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

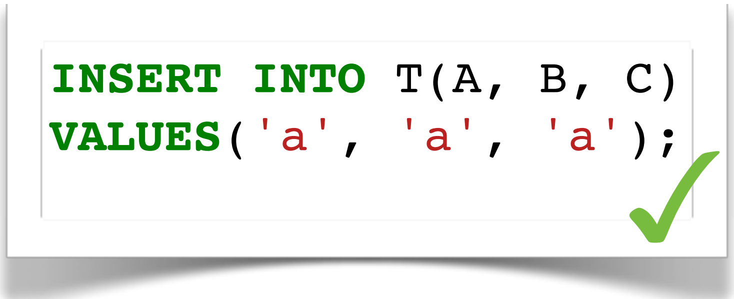
Database (mutated)



```sql
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```

Mutation Analysis

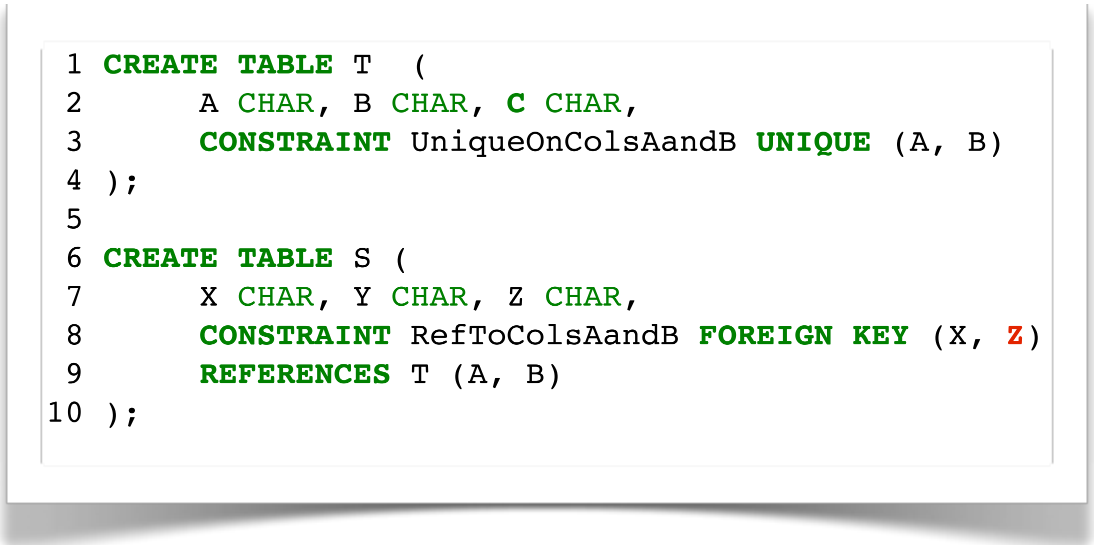
Insert Statements



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓

Database (mutated)

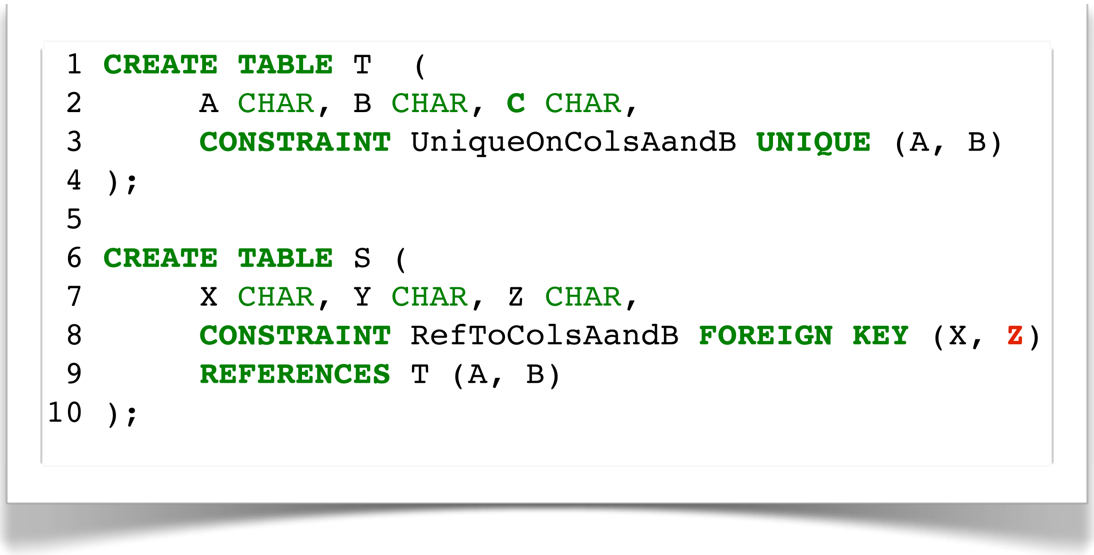


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```

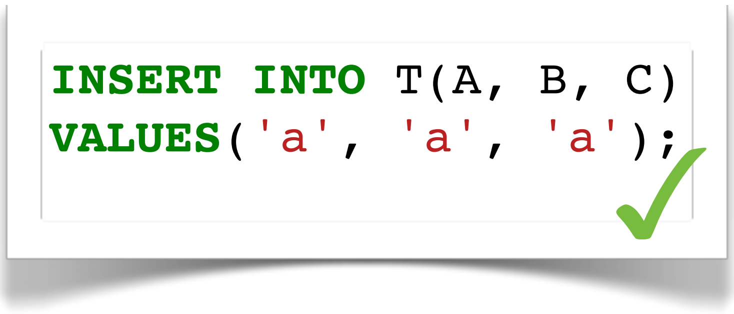
Mutation Analysis

Insert Statements

Database (mutated)



```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```

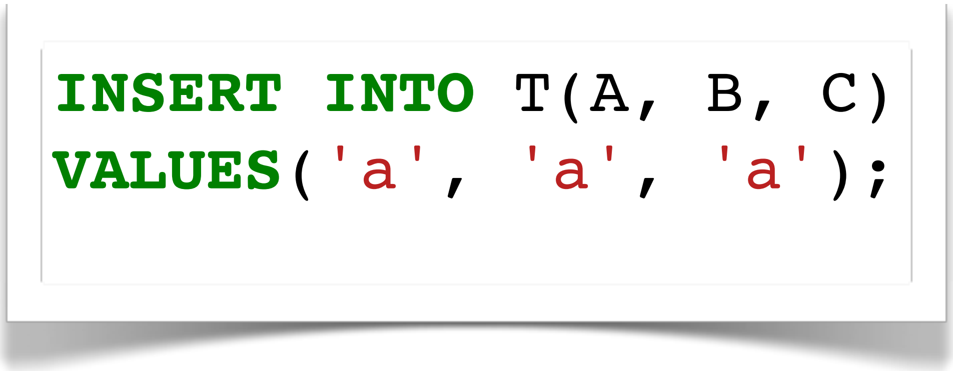


```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓

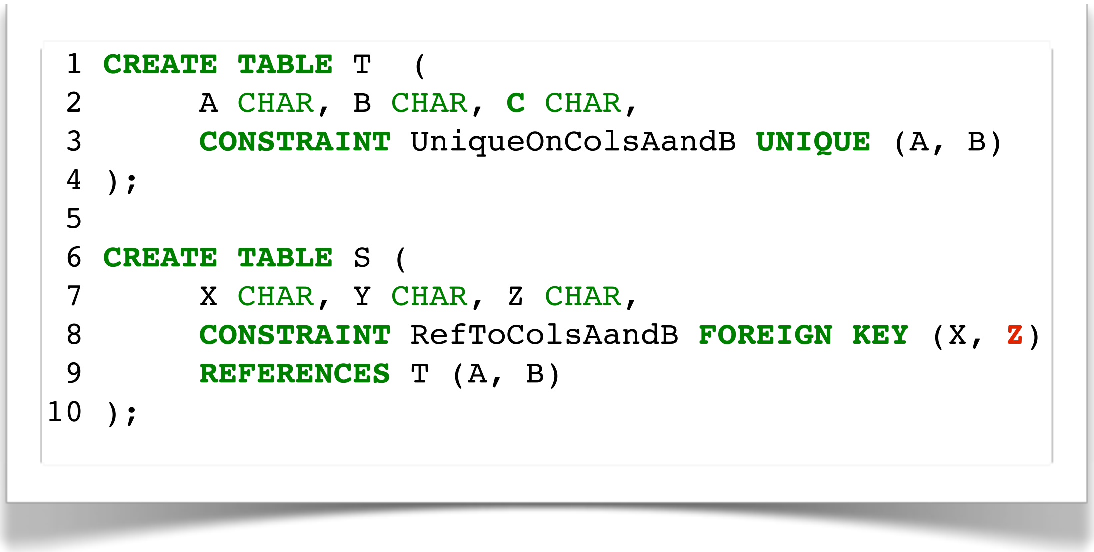
Mutation Analysis

Insert Statements

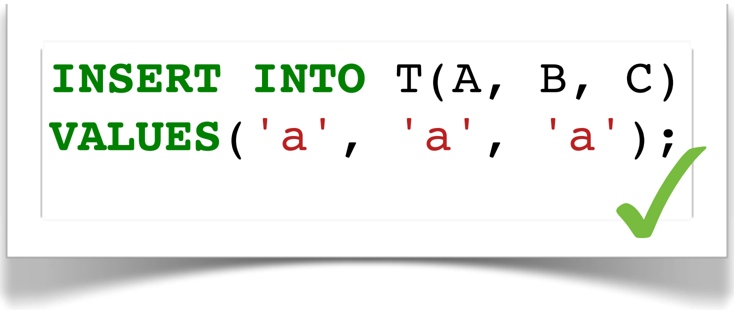


```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

Database (mutated)



```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```

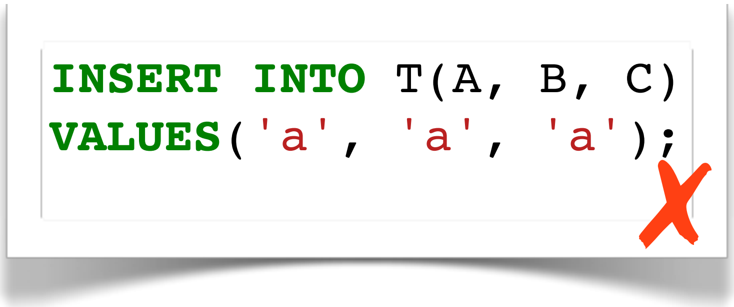


```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓

Mutation Analysis

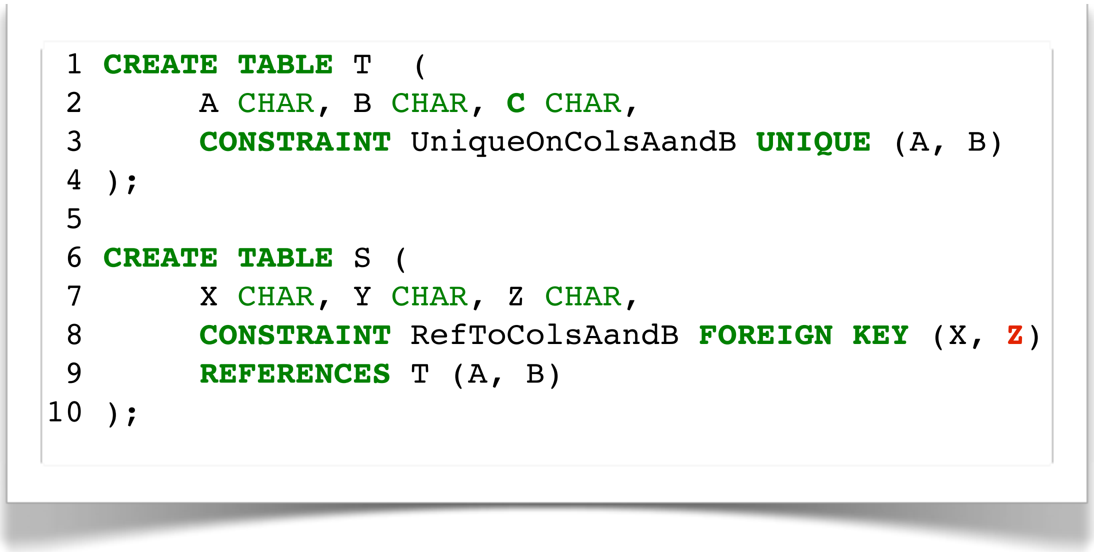
Insert Statements



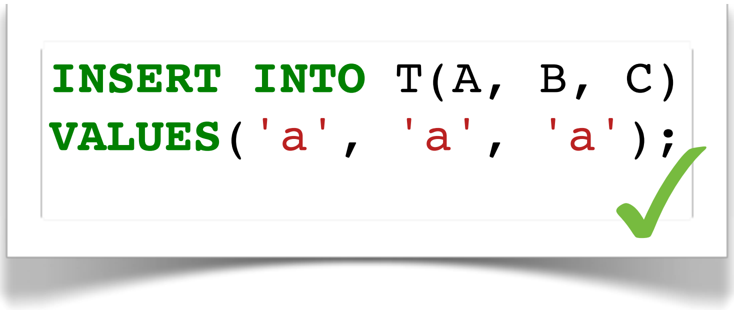
```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✗

Database (mutated)



```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```



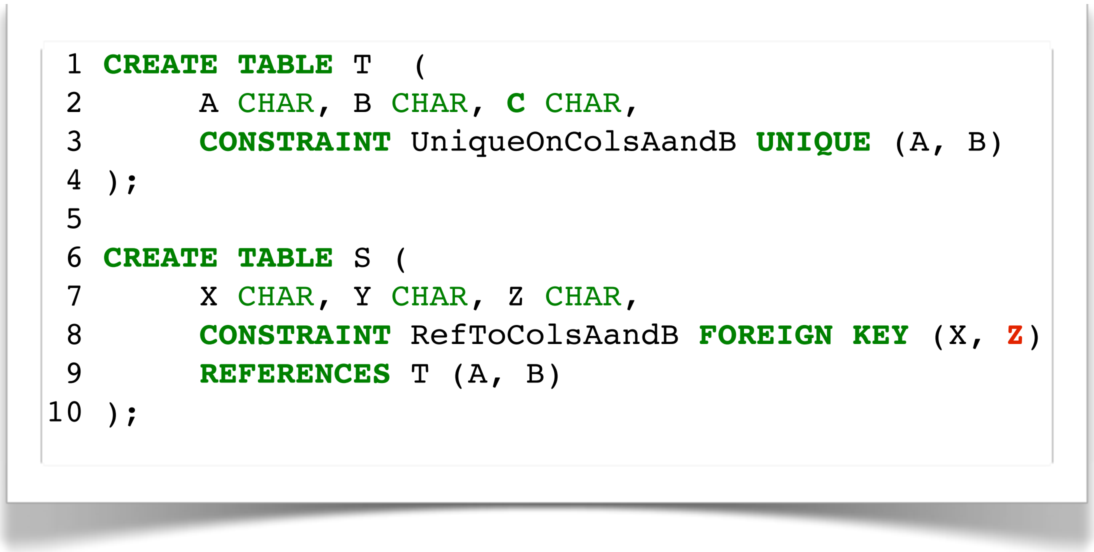
```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓

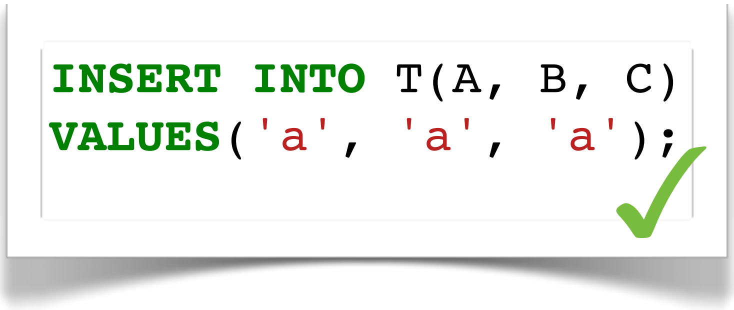
Mutation Analysis

Insert Statements

Database (mutated)

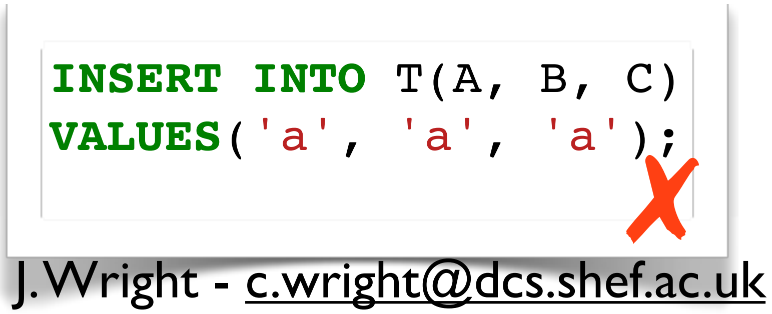


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓

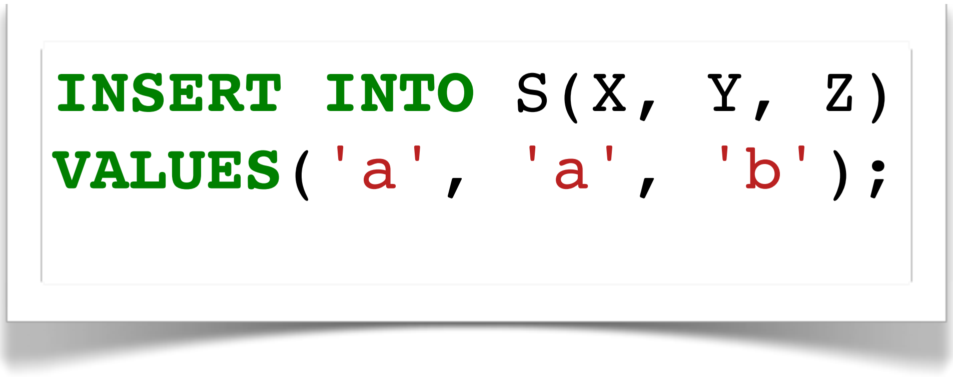


```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✗

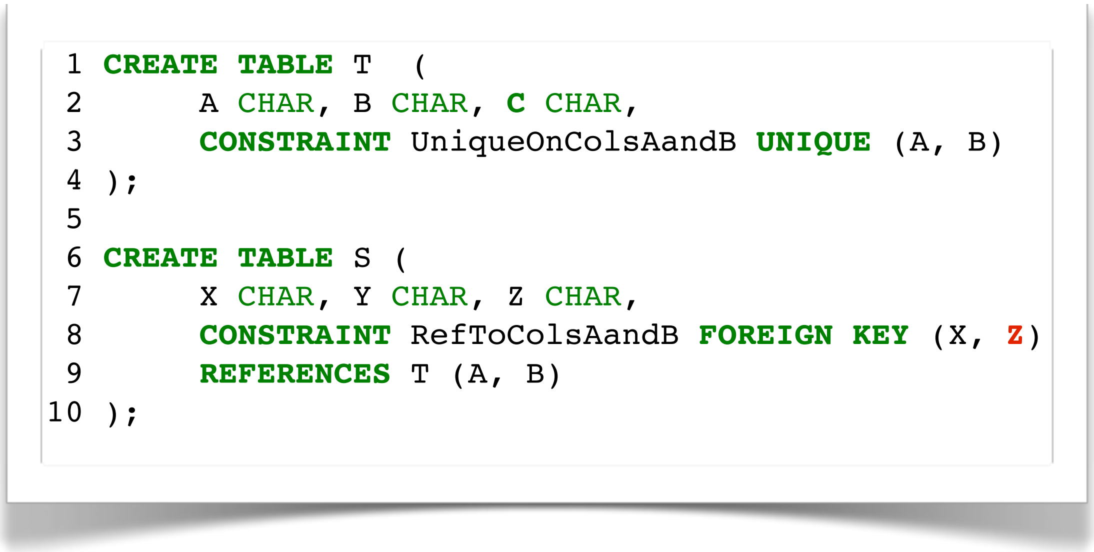
Mutation Analysis

Insert Statements

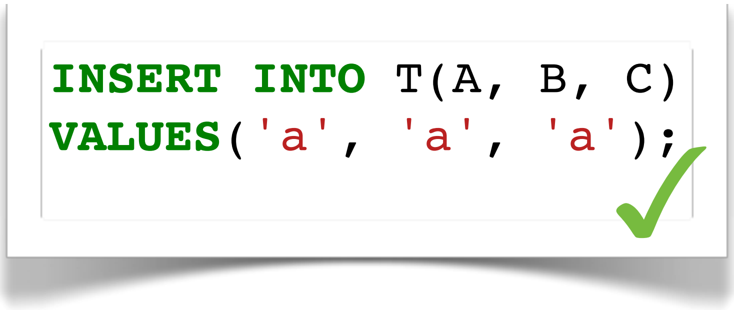


```
INSERT INTO S(X, Y, Z)
VALUES('a', 'a', 'b');
```

Database (mutated)

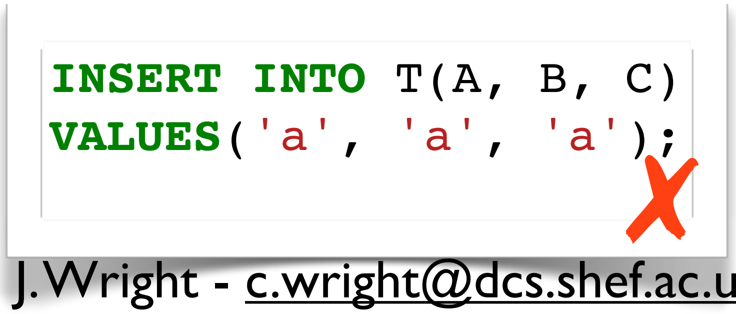


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓



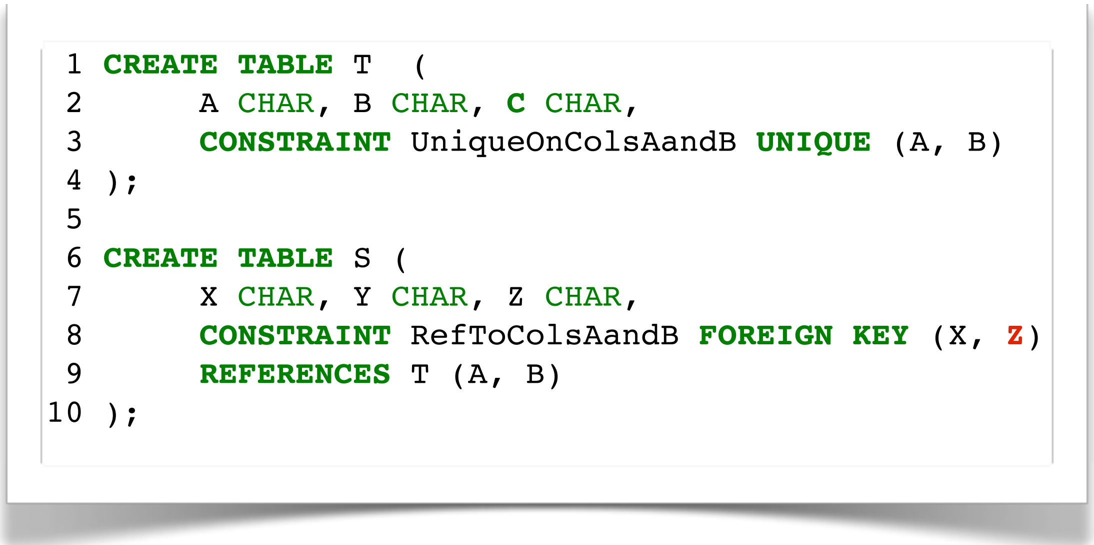
```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✗

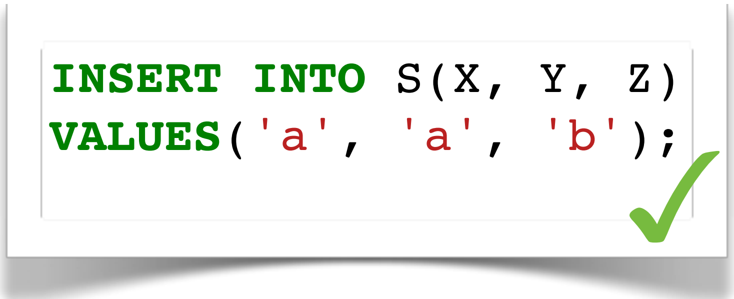
Mutation Analysis

Insert Statements

Database (mutated)

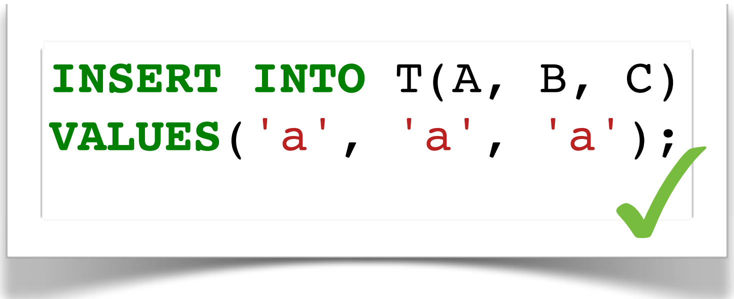


```sql
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```



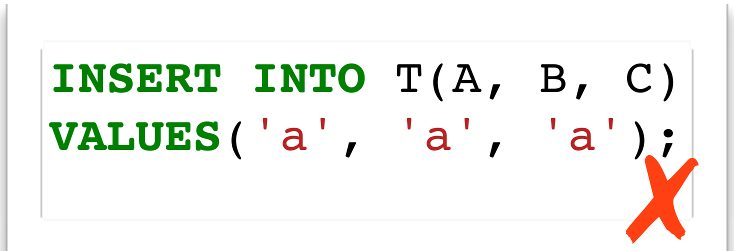
```sql
INSERT INTO S(X, Y, Z)
VALUES('a', 'a', 'b');
```

✓



```sql
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓



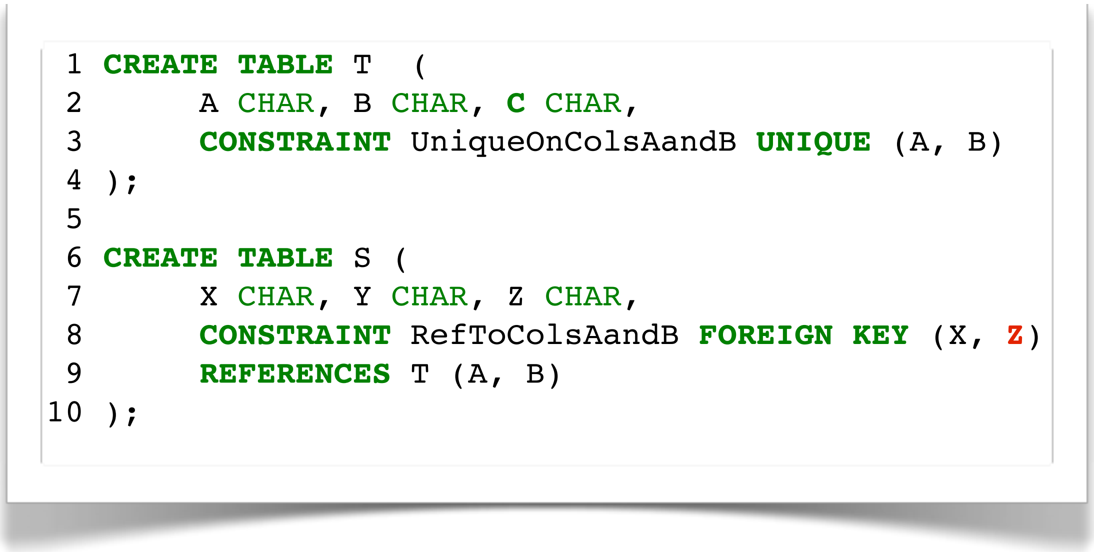
```sql
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✗

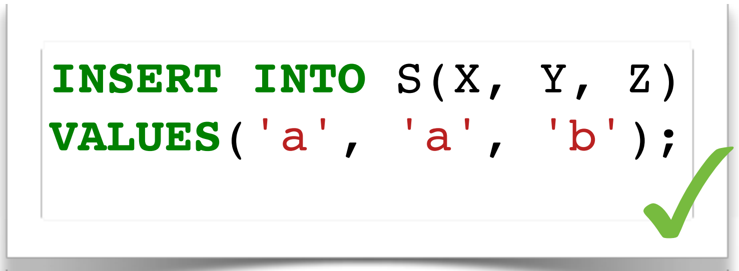
Mutation Analysis

Insert Statements

Database (mutated)

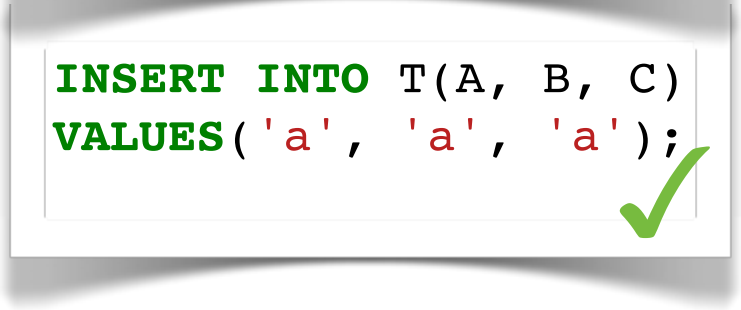


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```



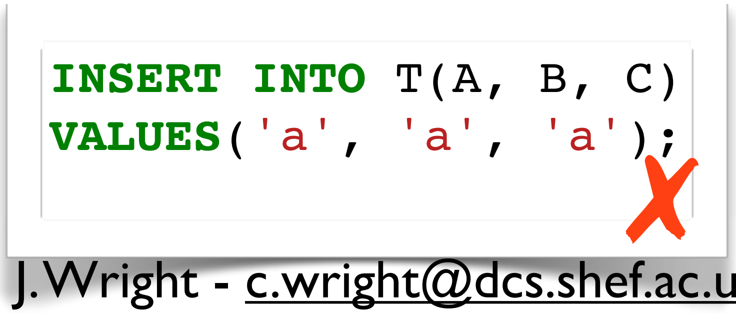
```
INSERT INTO S(X, Y, Z)
VALUES('a', 'a', 'b');
```

✓



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓



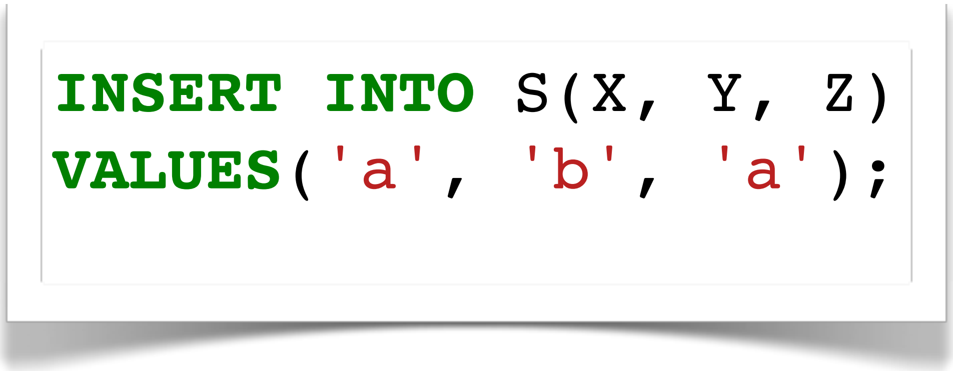
```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✗

Chris J. Wright - c.wright@dcs.shef.ac.uk

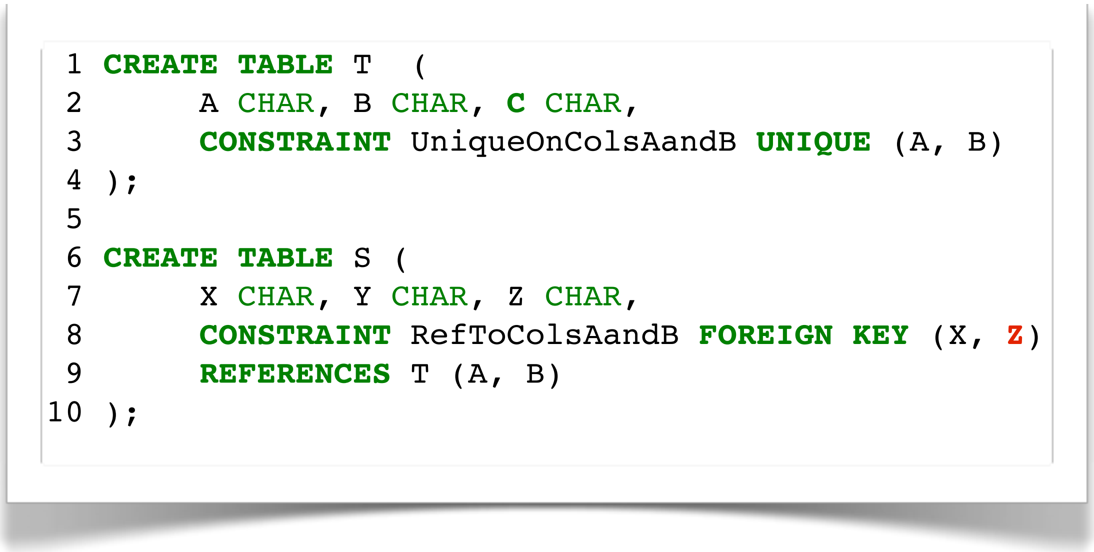
Mutation Analysis

Insert Statements

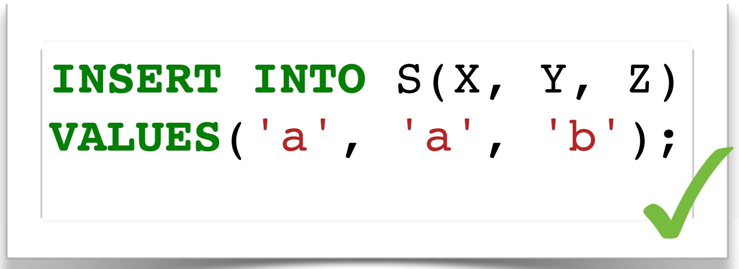


```
INSERT INTO S(X, Y, Z)
VALUES('a', 'b', 'a');
```

Database (mutated)

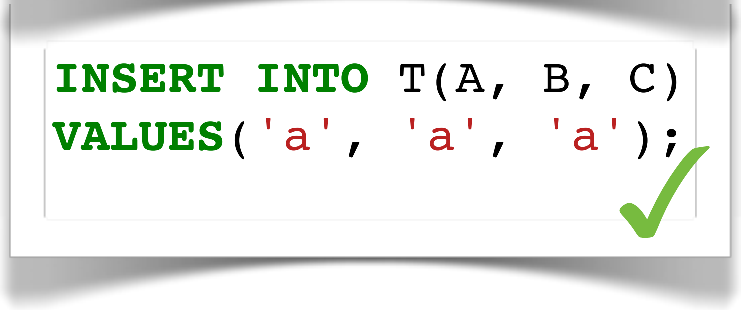


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```



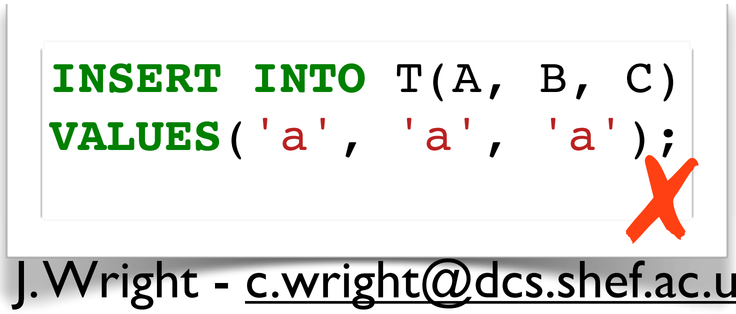
```
INSERT INTO S(X, Y, Z)
VALUES('a', 'a', 'b');
```

✓



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓



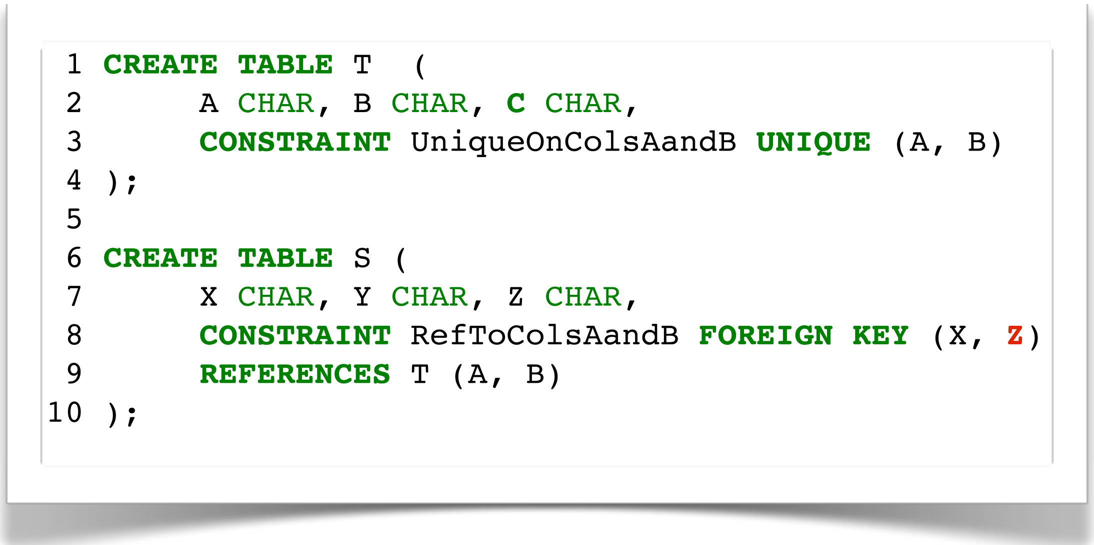
```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✗

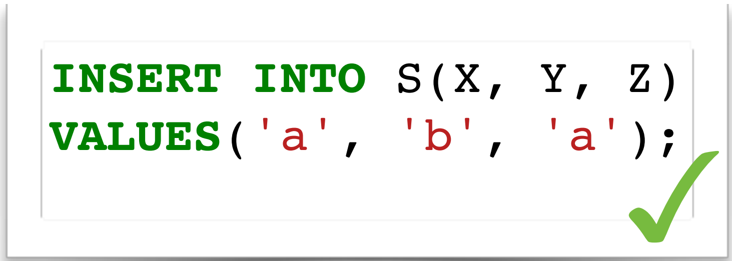
Mutation Analysis

Insert Statements

Database (mutated)

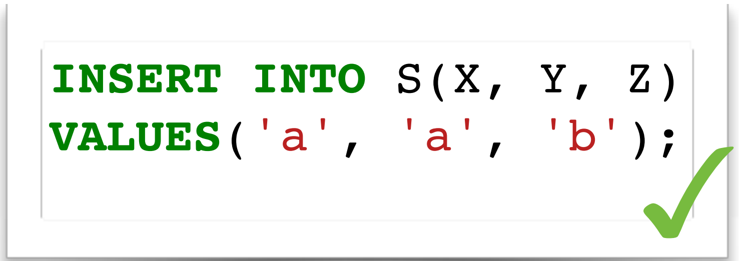


```
CREATE TABLE T  (
      A CHAR, B CHAR, C CHAR,
      CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
      X CHAR, Y CHAR, Z CHAR,
      CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
      REFERENCES T (A, B)
);
```



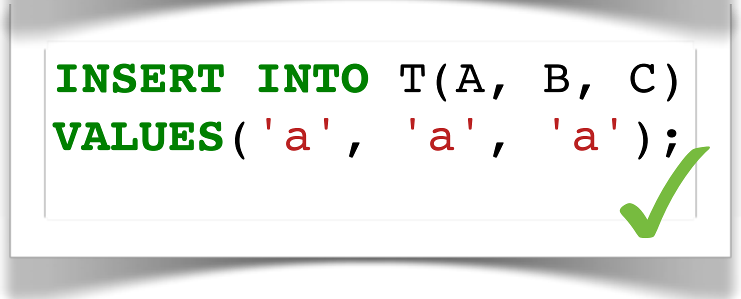
```
INSERT INTO S(X, Y, Z)
VALUES('a', 'b', 'a');
```

✓



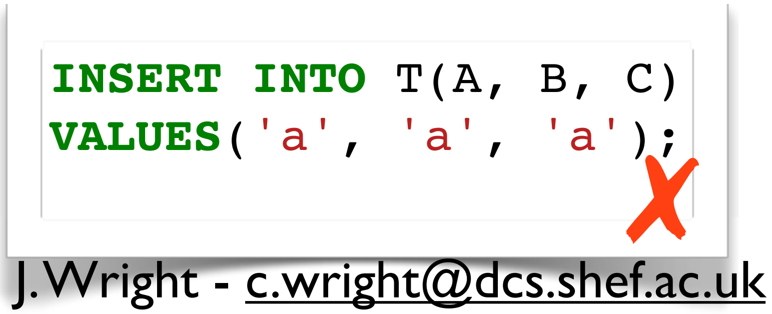
```
INSERT INTO S(X, Y, Z)
VALUES('a', 'a', 'b');
```

✓



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓



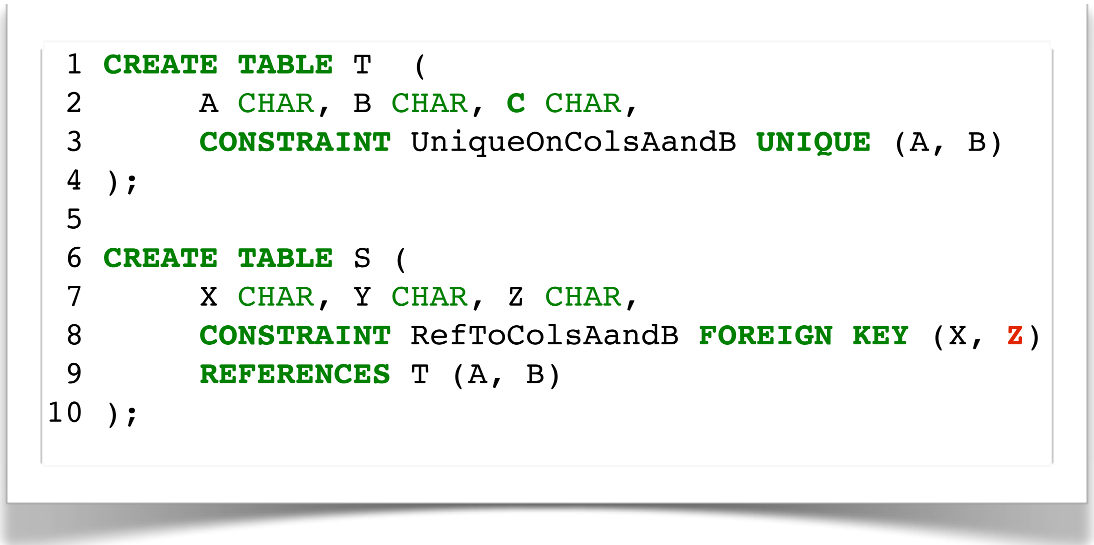
```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✗

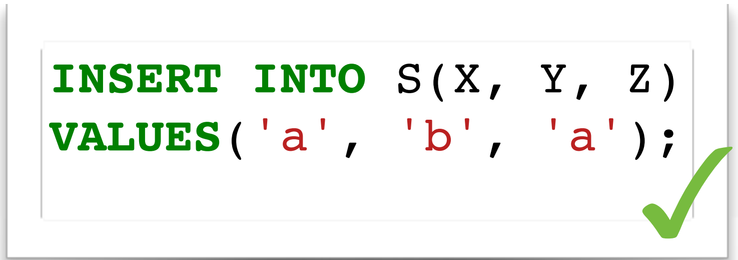
Mutation Analysis

Insert Statements

Database (mutated)

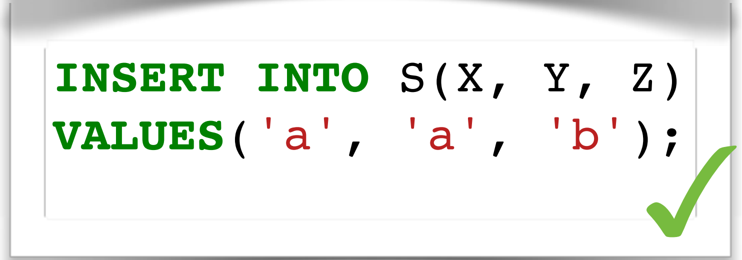


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```



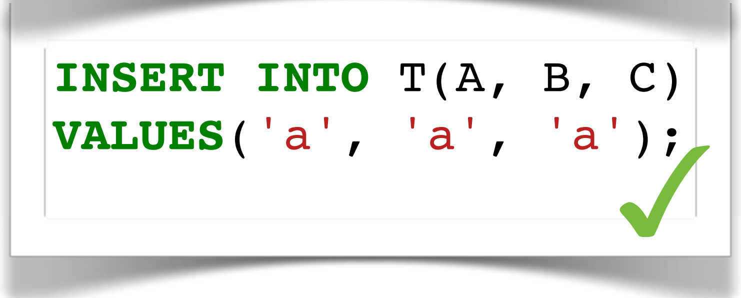
```
INSERT INTO S(X, Y, Z)
VALUES('a', 'b', 'a');
```

✓



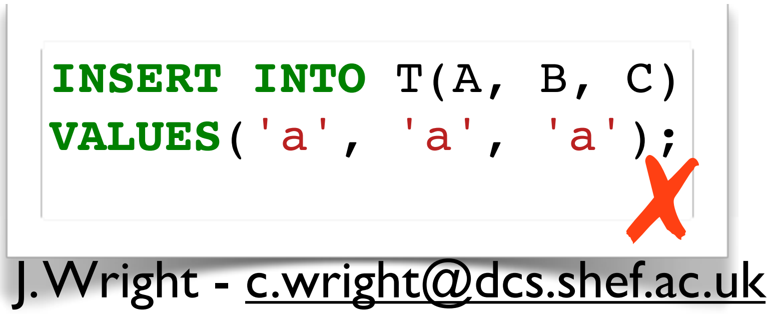
```
INSERT INTO S(X, Y, Z)
VALUES('a', 'a', 'b');
```

✓



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓



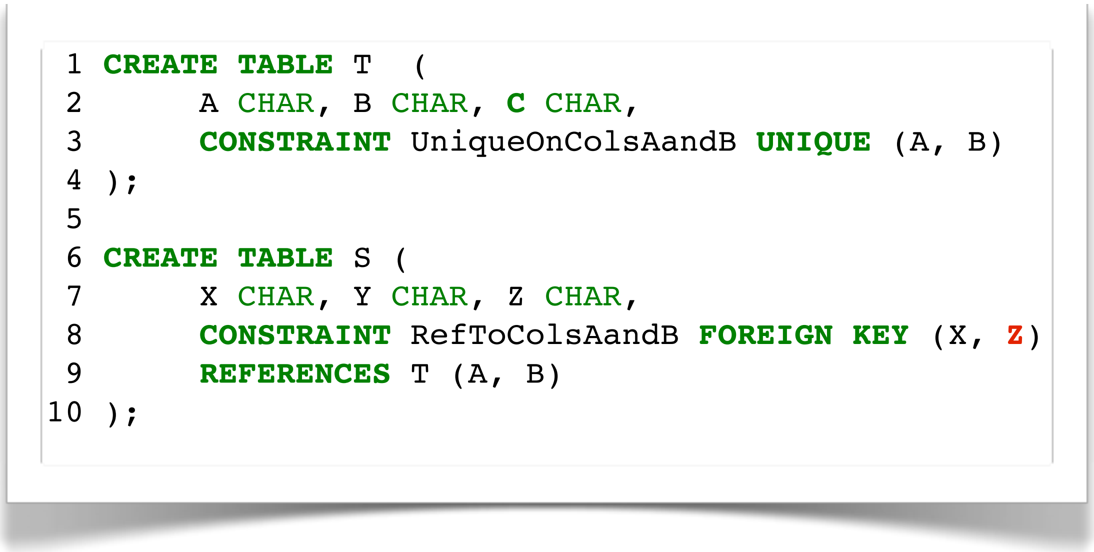
```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✗

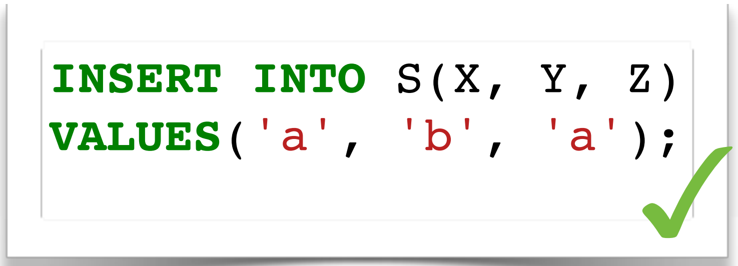
Mutation Analysis

Insert Statements

Database (mutated)

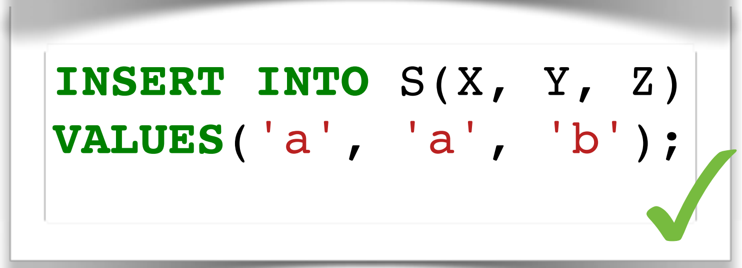


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```



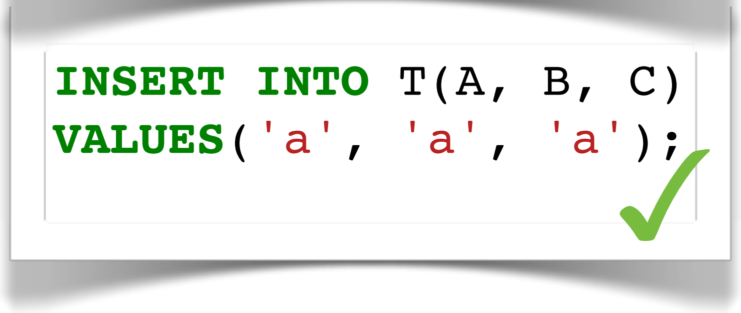
```
INSERT INTO S(X, Y, Z)
VALUES('a', 'b', 'a');
```

✓



```
INSERT INTO S(X, Y, Z)
VALUES('a', 'a', 'b');
```

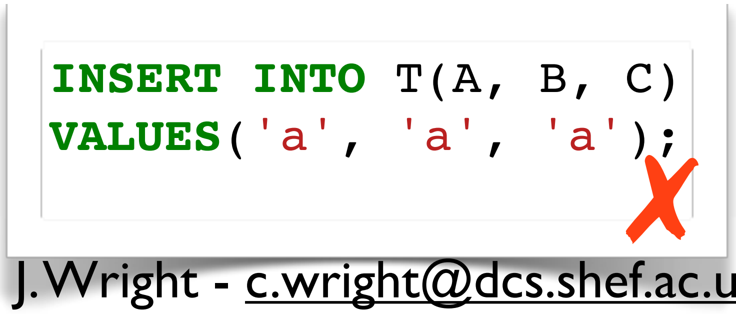
✓



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓

Results are different



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

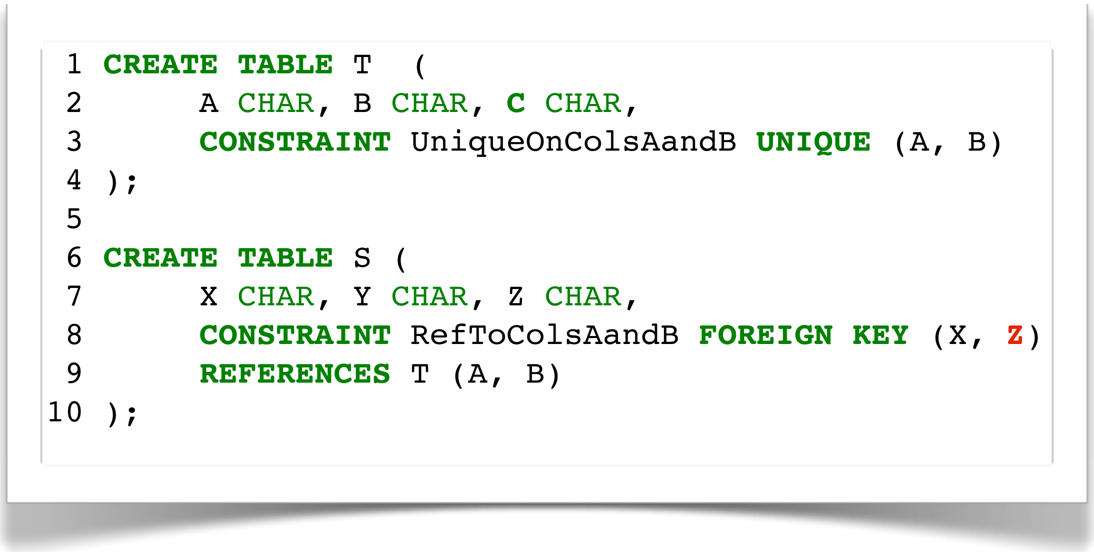
✗

Chris J. Wright - c.wright@dcs.shef.ac.uk

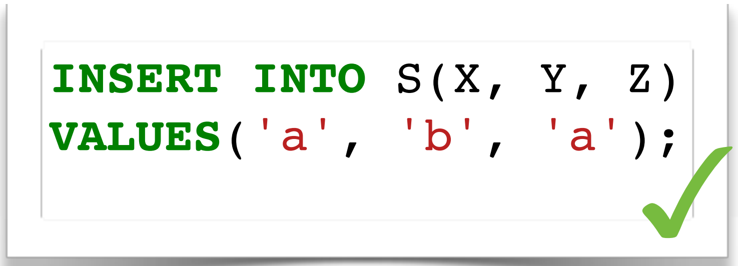
Mutation Analysis

Insert Statements

Database (mutated)

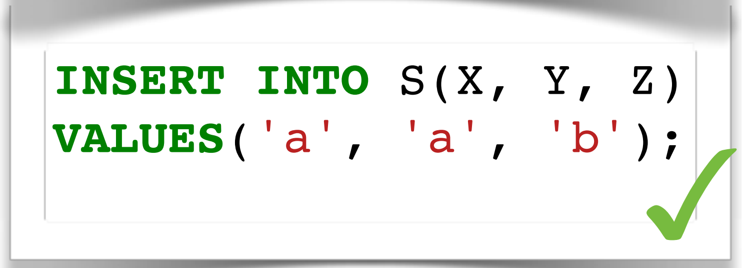


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```



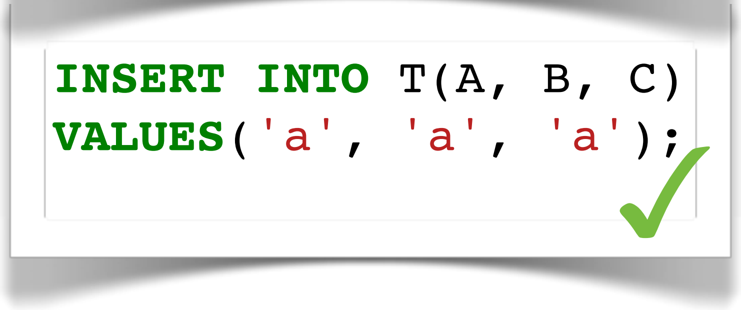
```
INSERT INTO S(X, Y, Z)
VALUES('a', 'b', 'a');
```

✓



```
INSERT INTO S(X, Y, Z)
VALUES('a', 'a', 'b');
```

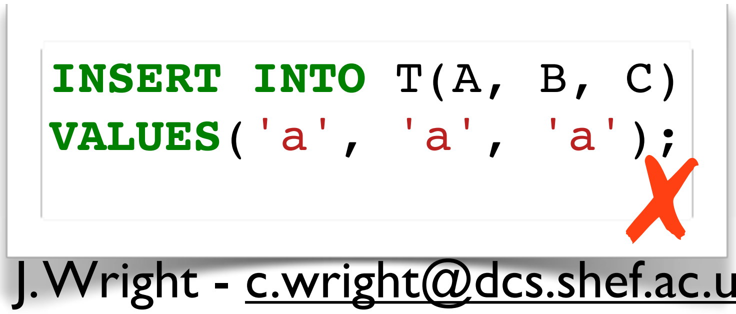
✓



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✓

Mutant is killed



```
INSERT INTO T(A, B, C)
VALUES('a', 'a', 'a');
```

✗

Mutation Operators

Mutation Operators

Primary Key

Mutation Operators

Primary Key

Not Null

Mutation Operators

Primary Key

Not Null

Check

Mutation Operators

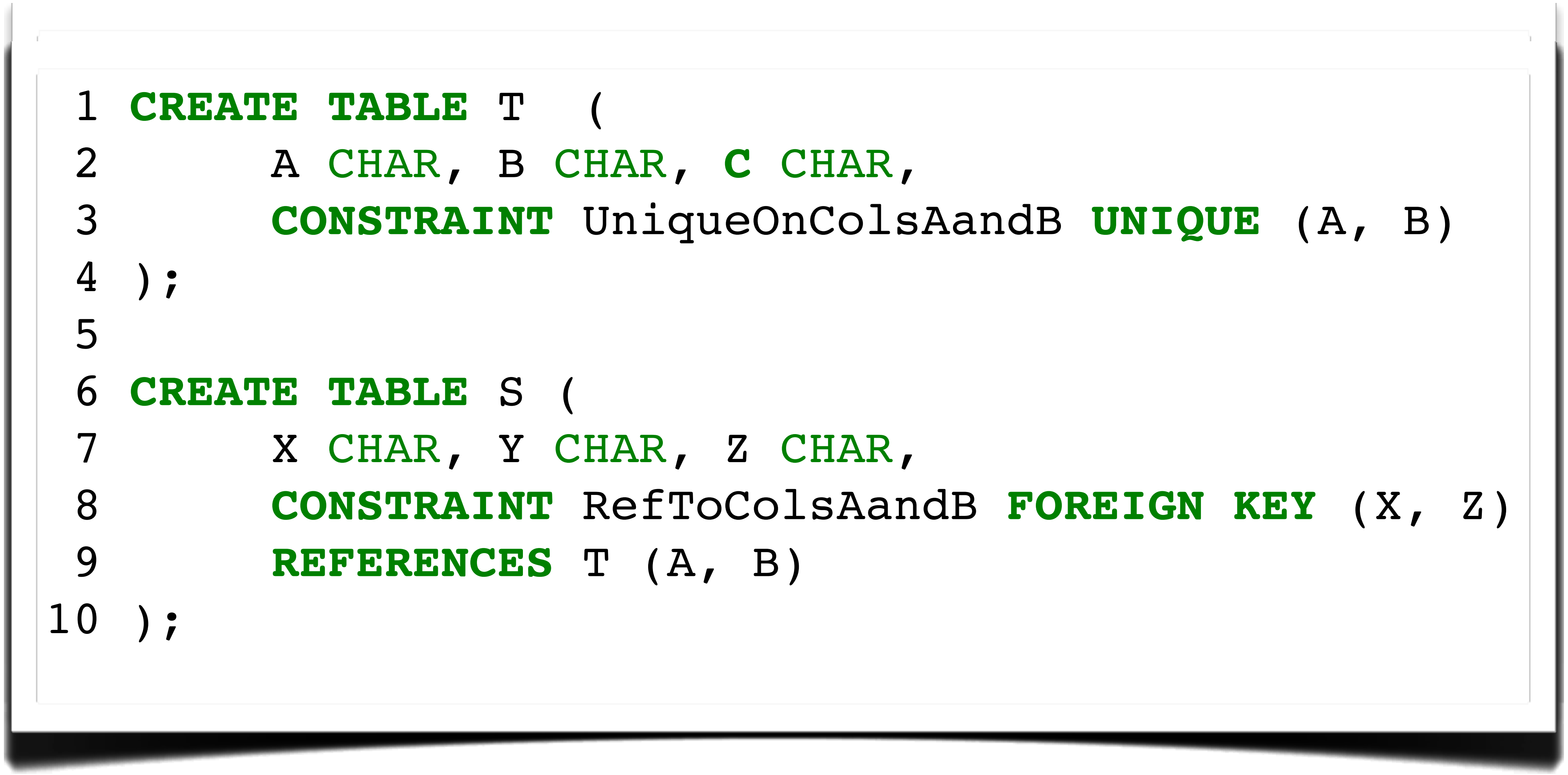
Primary Key

Not Null

Check

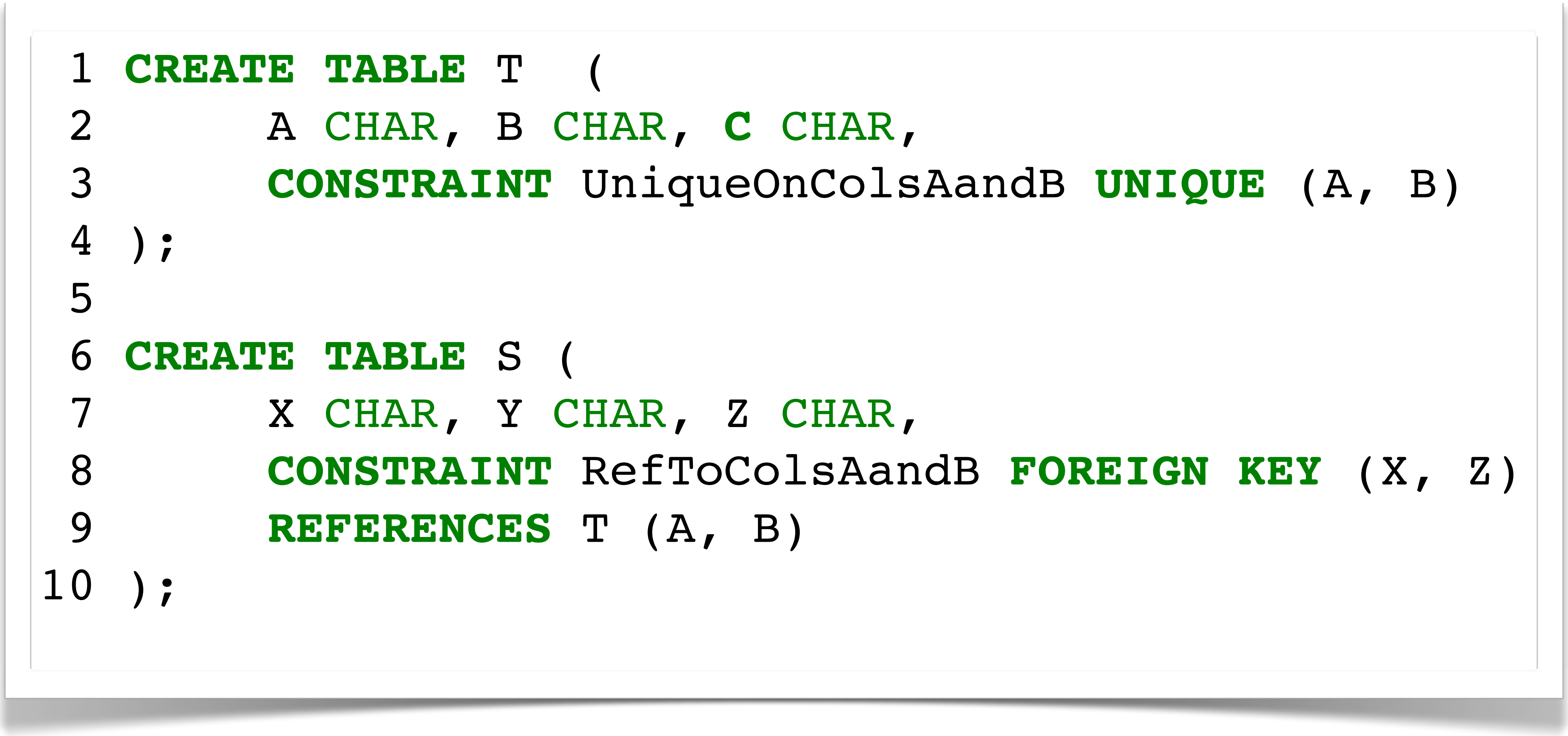
Unique

The Problem?

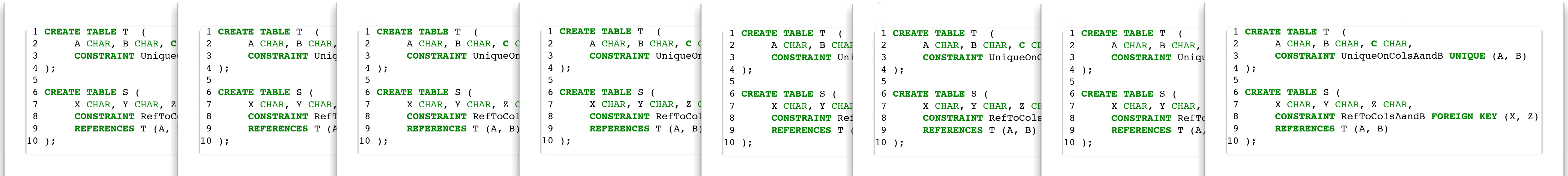


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```

The Problem?



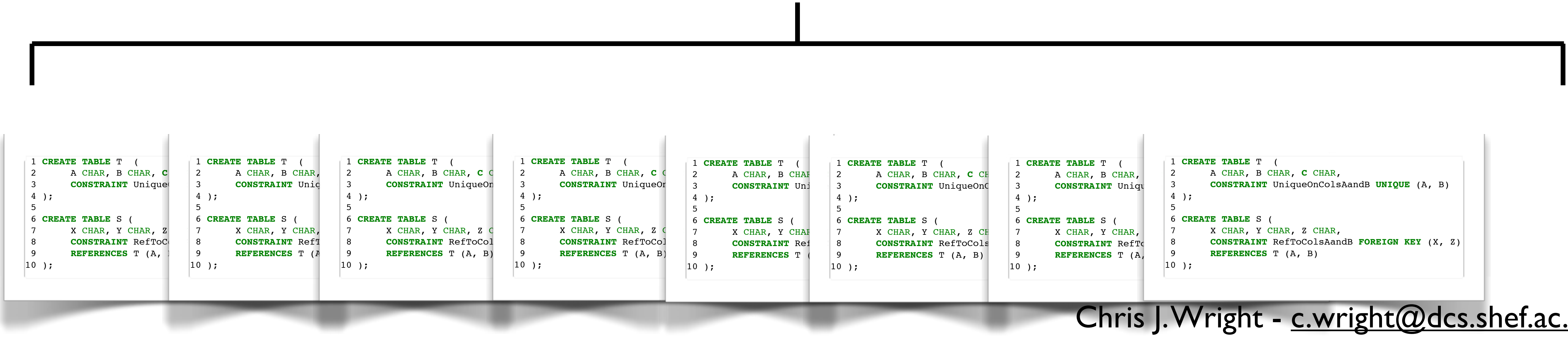
```sql
CREATE TABLE T  (
      A CHAR, B CHAR, C CHAR,
      CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
      X CHAR, Y CHAR, Z CHAR,
      CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
      REFERENCES T (A, B)
);
```



```sql
CREATE TABLE T  (
      A CHAR, B CHAR, C CHAR,
      CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
      X CHAR, Y CHAR, Z CHAR,
      CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
      REFERENCES T (A, B)
);
```

The Problem?

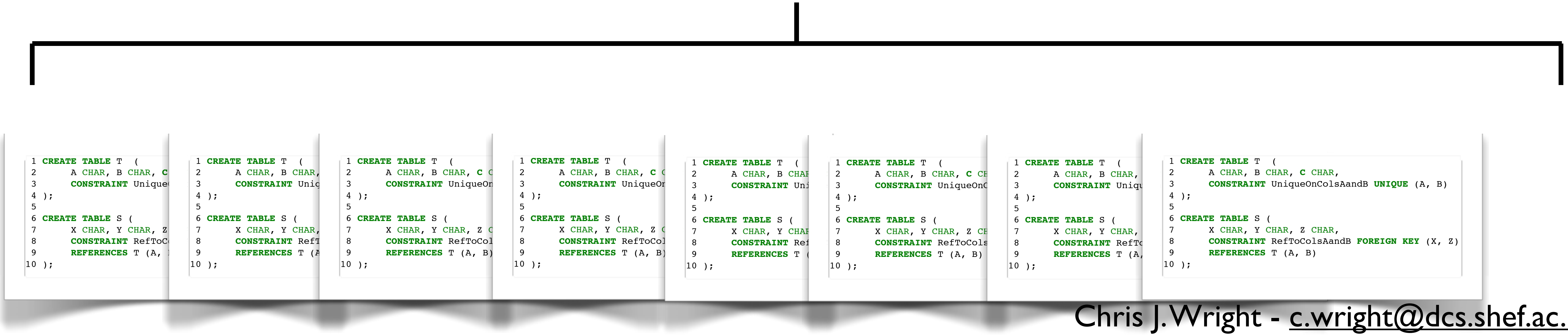
Many mutants
to analyse



```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```

The Problem?

A time consuming process



```
CREATE TABLE T  (
      A CHAR, B CHAR, C CHAR,
      CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
      X CHAR, Y CHAR, Z CHAR,
      CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
      REFERENCES T (A, B)
);
```

The Solution?

The Solution?

Mutant Schemata

The Solution?

Mutant Schemata

Combine mutants into a 'meta-mutant'

The Solution?

Mutant Schemata

Combine mutants into a 'meta-mutant'

Parallelisation

The Solution?

Mutant Schemata

Combine mutants into a ‘meta-mutant’

Parallelisation

Analyse multiple mutants simultaneously

Mutation Analysis Approaches

Mutant Representation

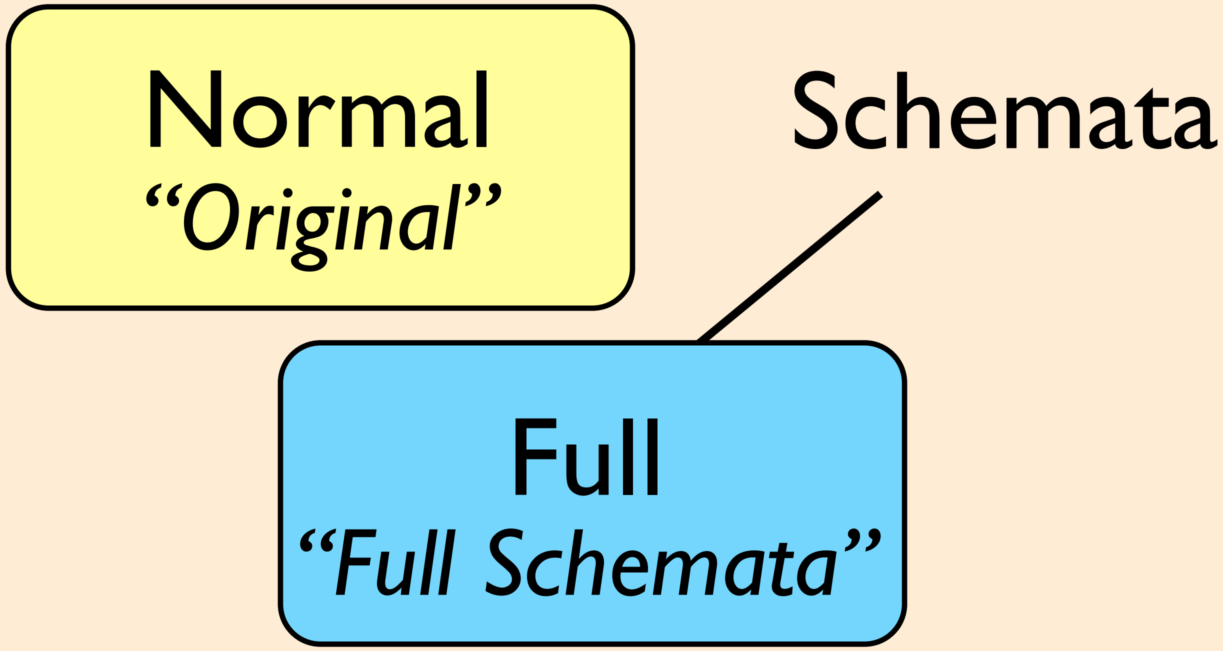
Normal

"Original"

Parallelisation Strategy

Mutation Analysis Approaches

Mutant Representation



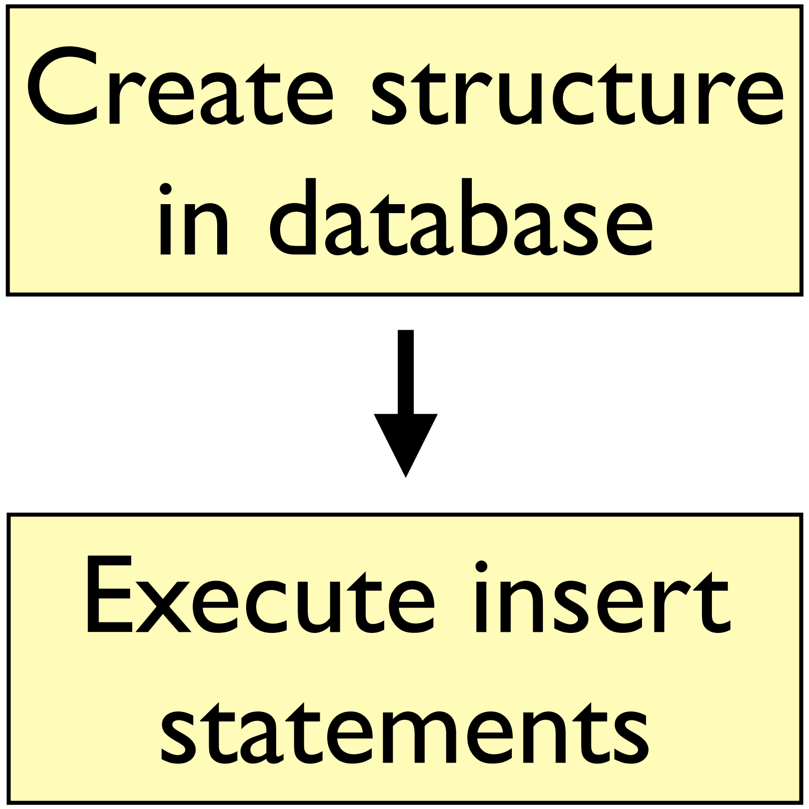
Parallelisation Strategy

Original Approach

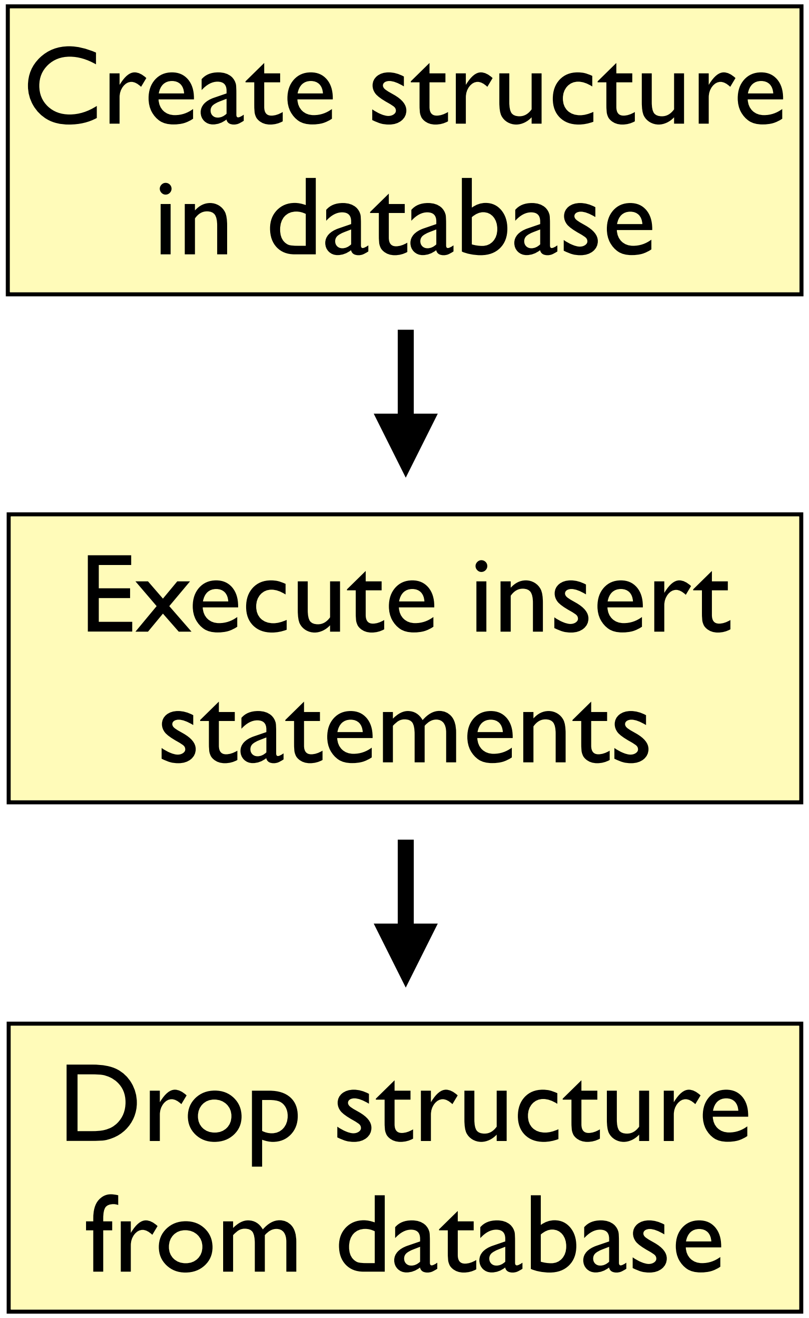
Original Approach

Create structure in database

Original Approach



Original Approach

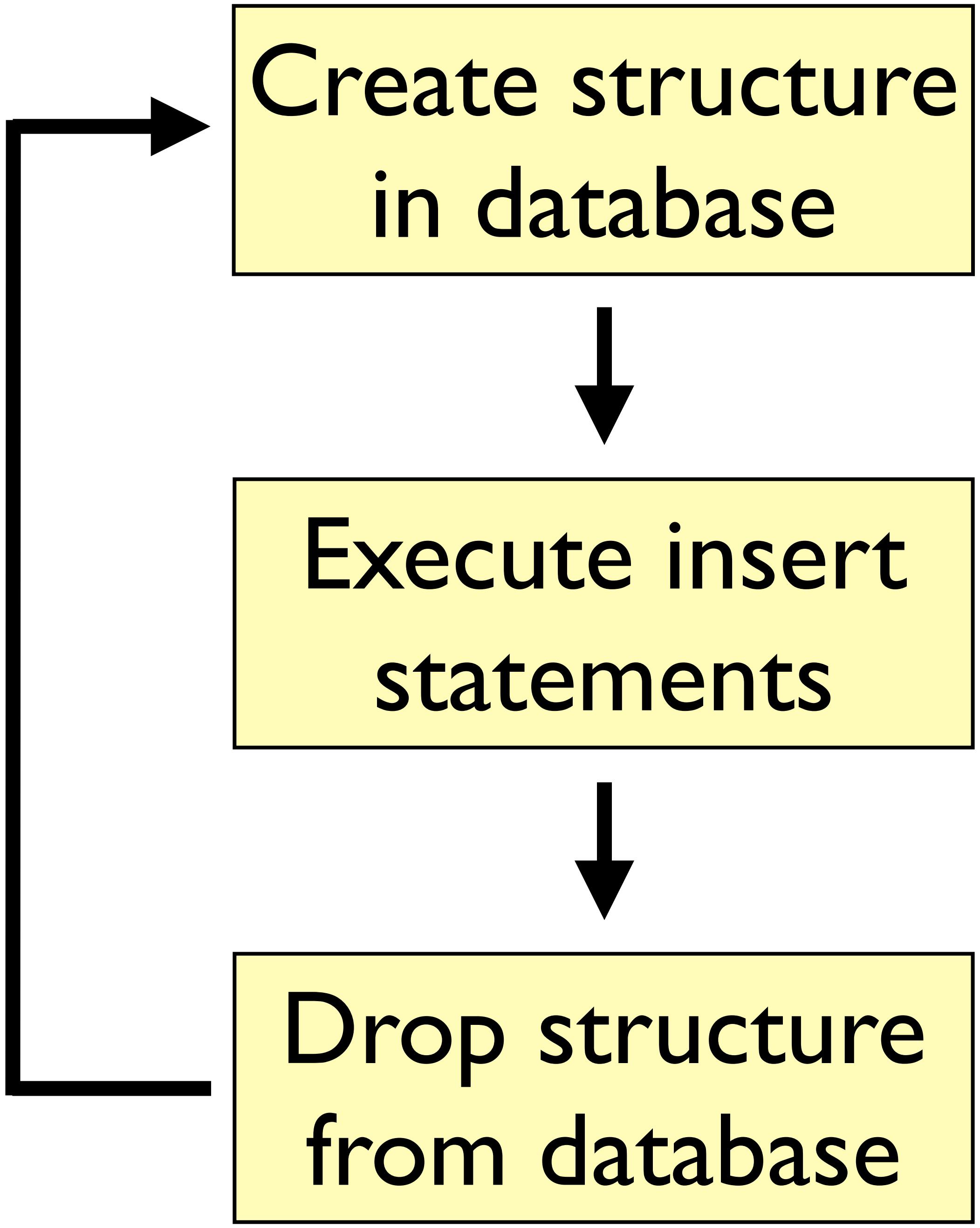


Create structure in database

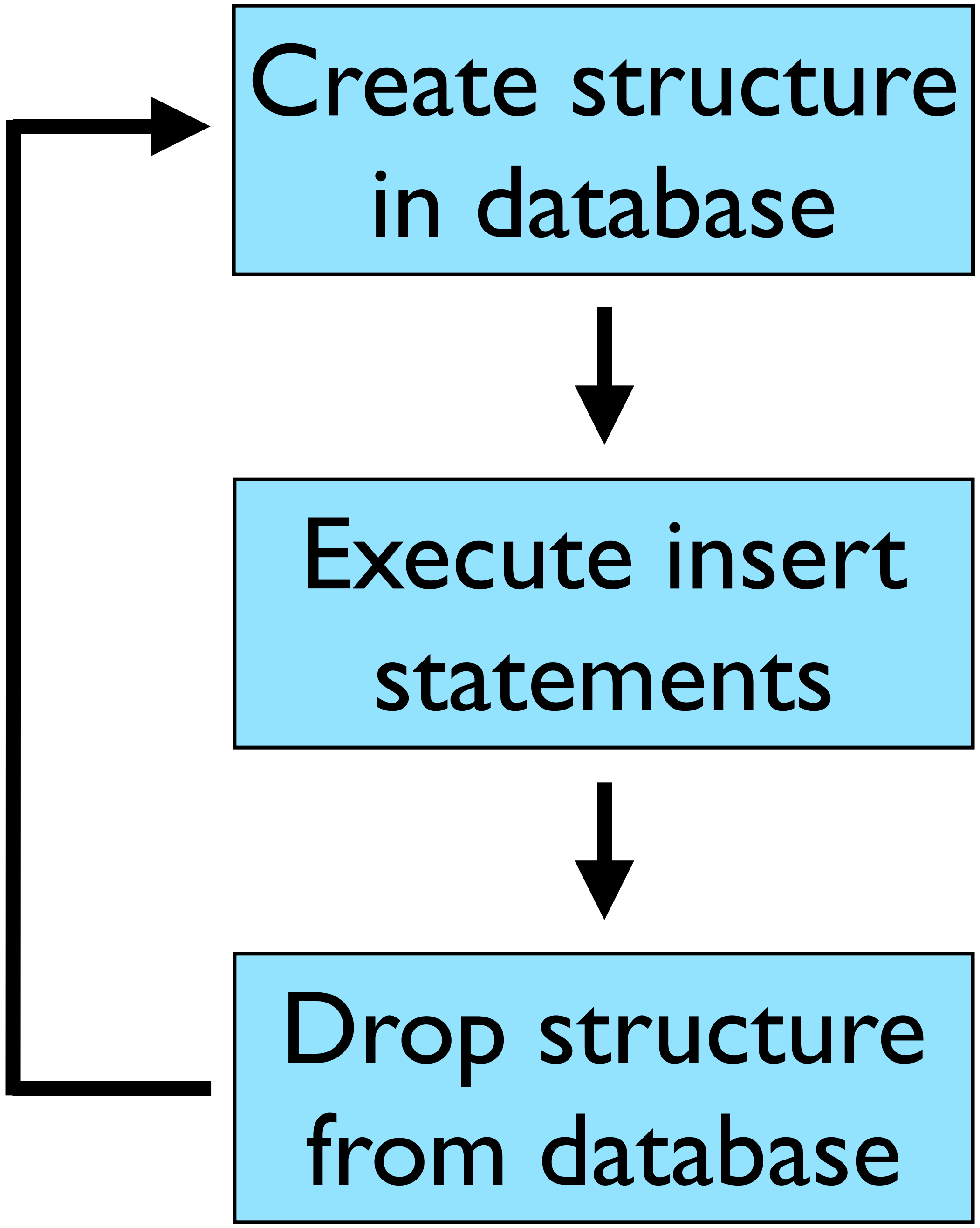
Execute insert statements

Drop structure from database

Original Approach


Create structure in database
Execute insert statements
Drop structure from database

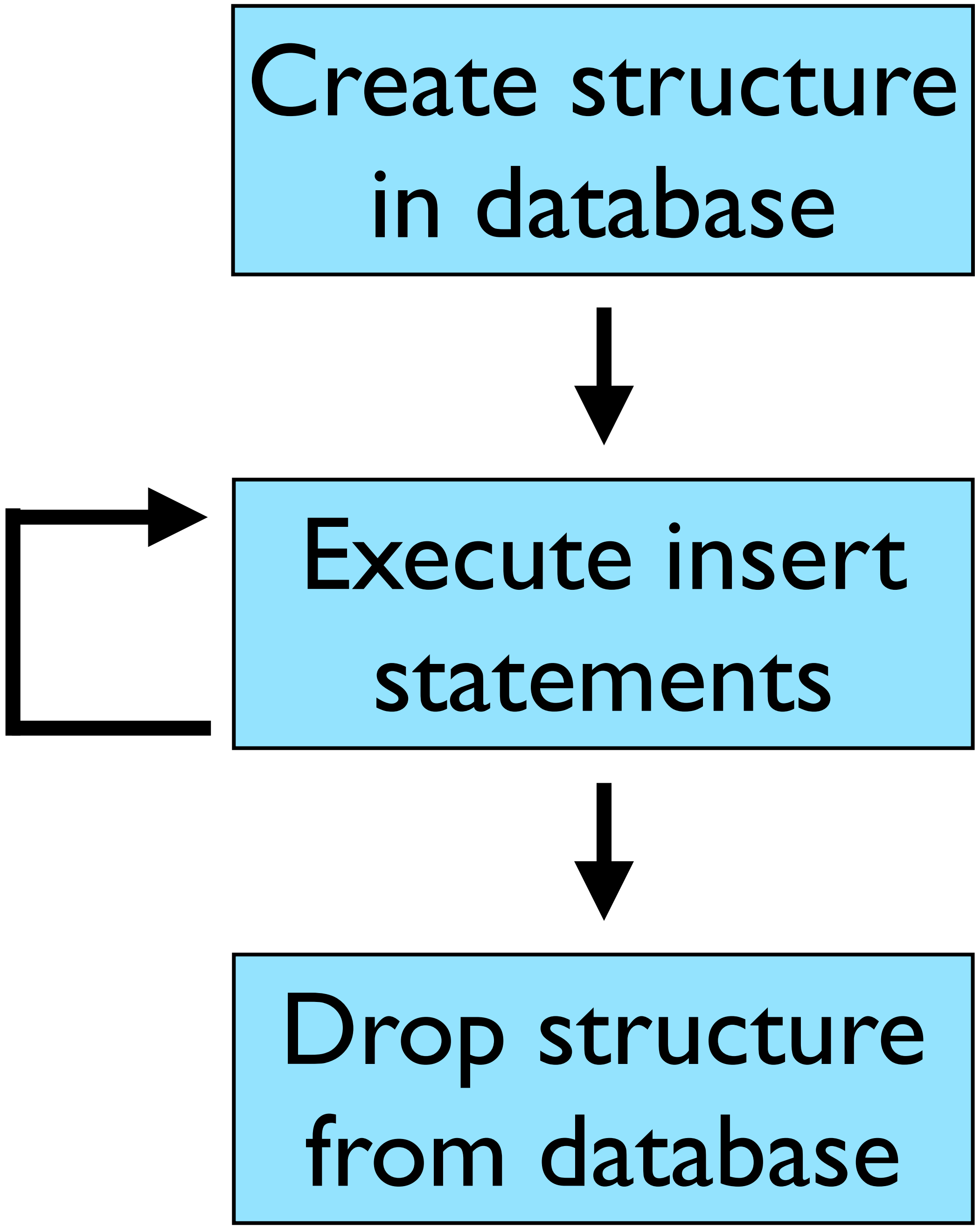
Mutant Schemata Approach


Create structure in database

Execute insert statements

Drop structure from database

Mutant Schemata Approach

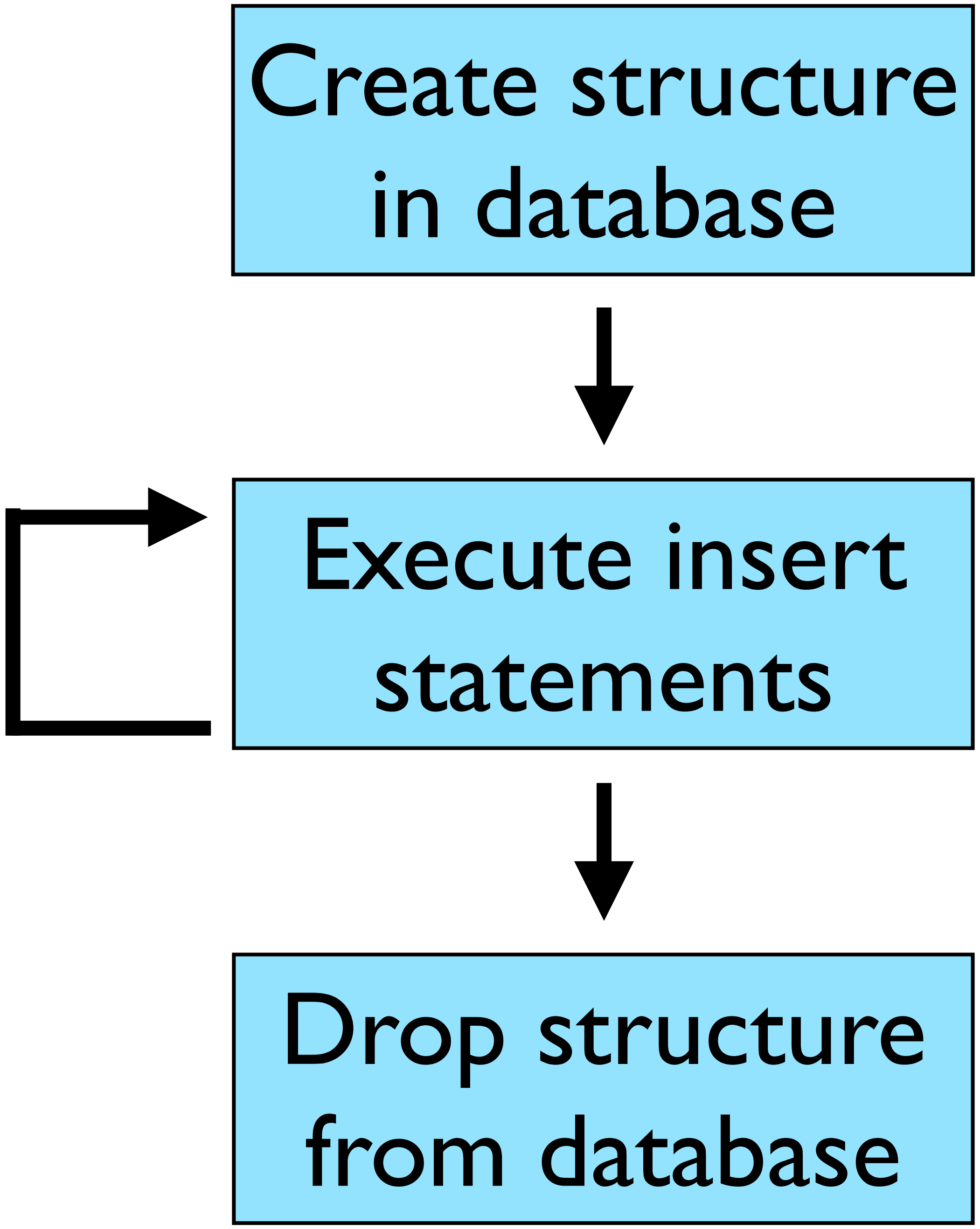


Create structure in database

Execute insert statements

Drop structure from database

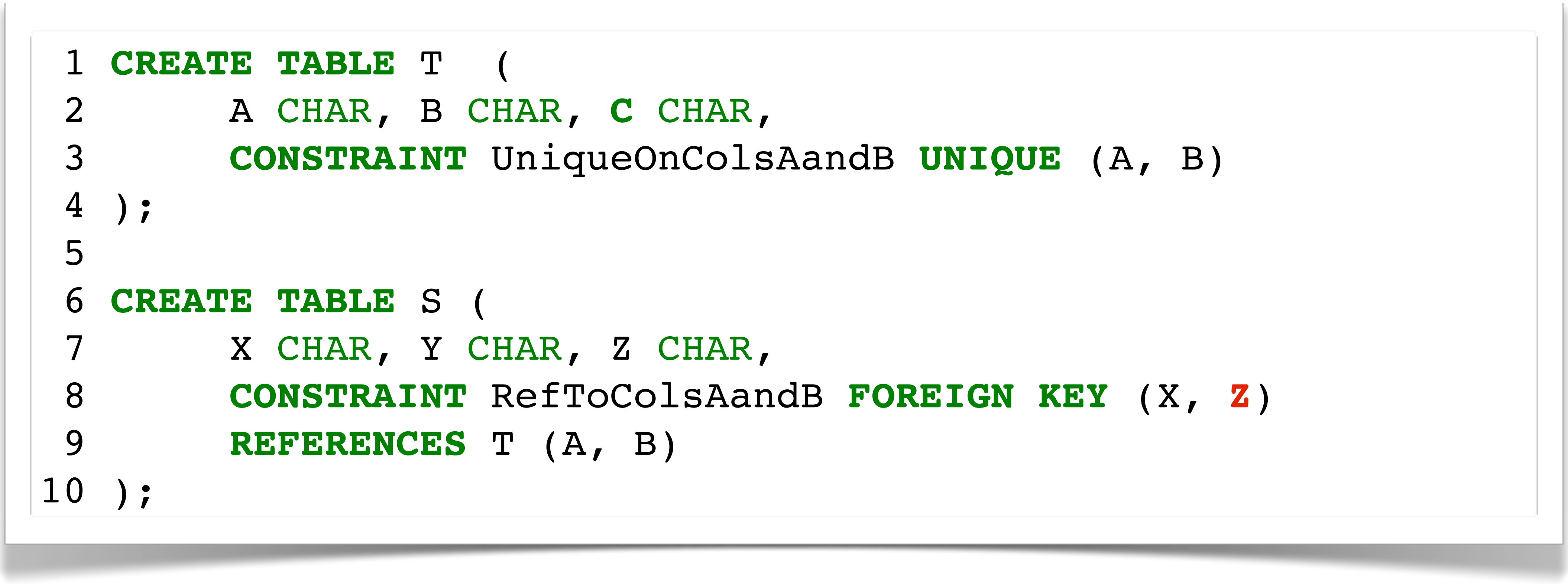
Mutant Schemata Approach


Create structure in database
Execute insert statements
Drop structure from database

Reduce time creating/dropping database

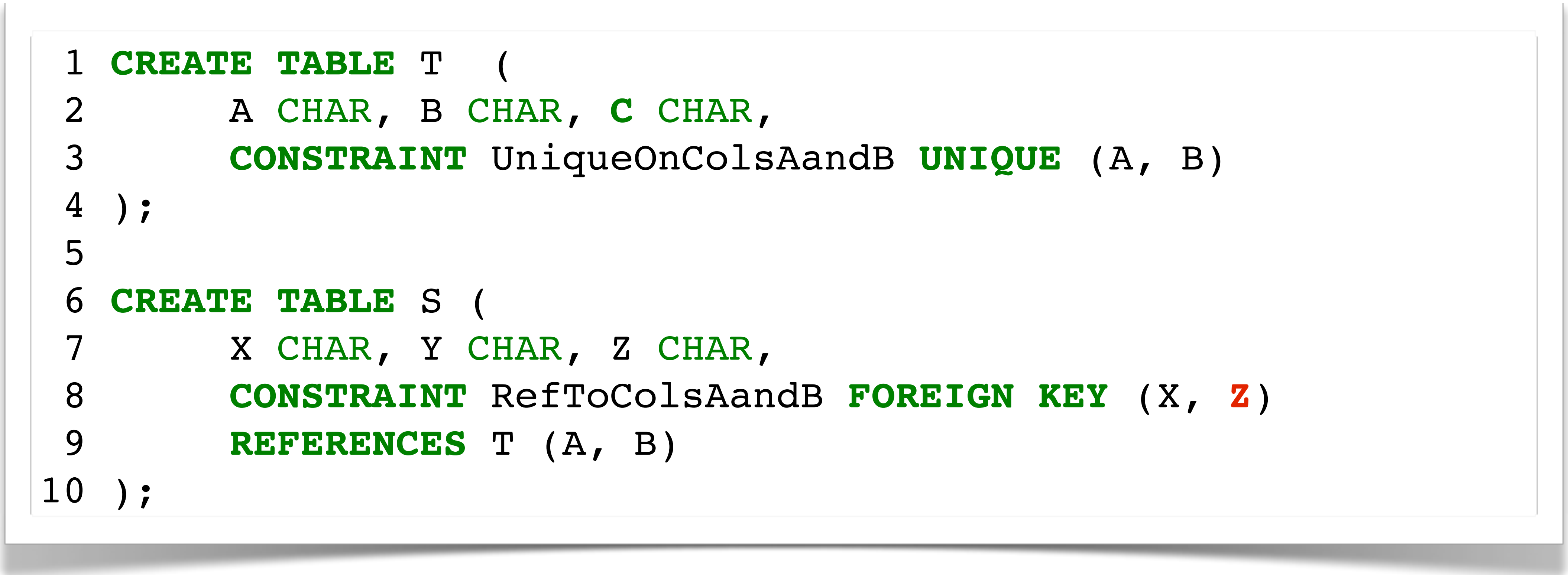
Full Schemata Approach

Full Schemata Approach

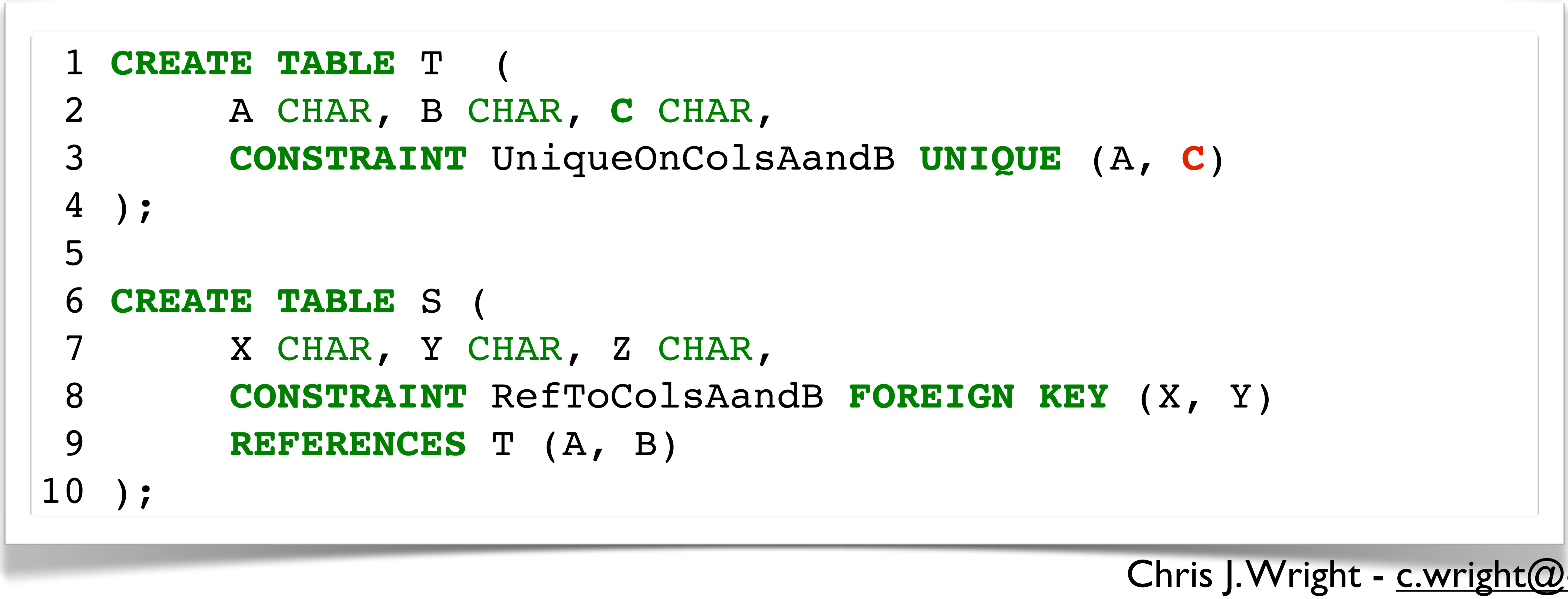


```sql
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```

Full Schemata Approach

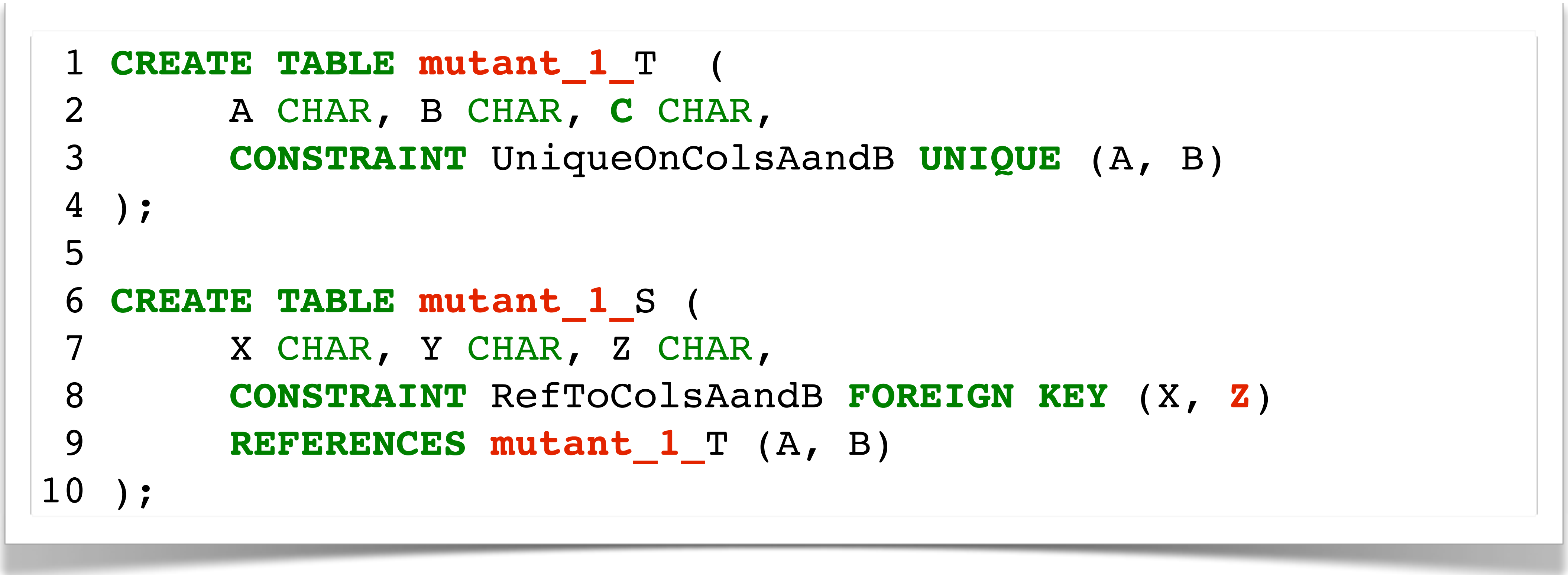


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);
```

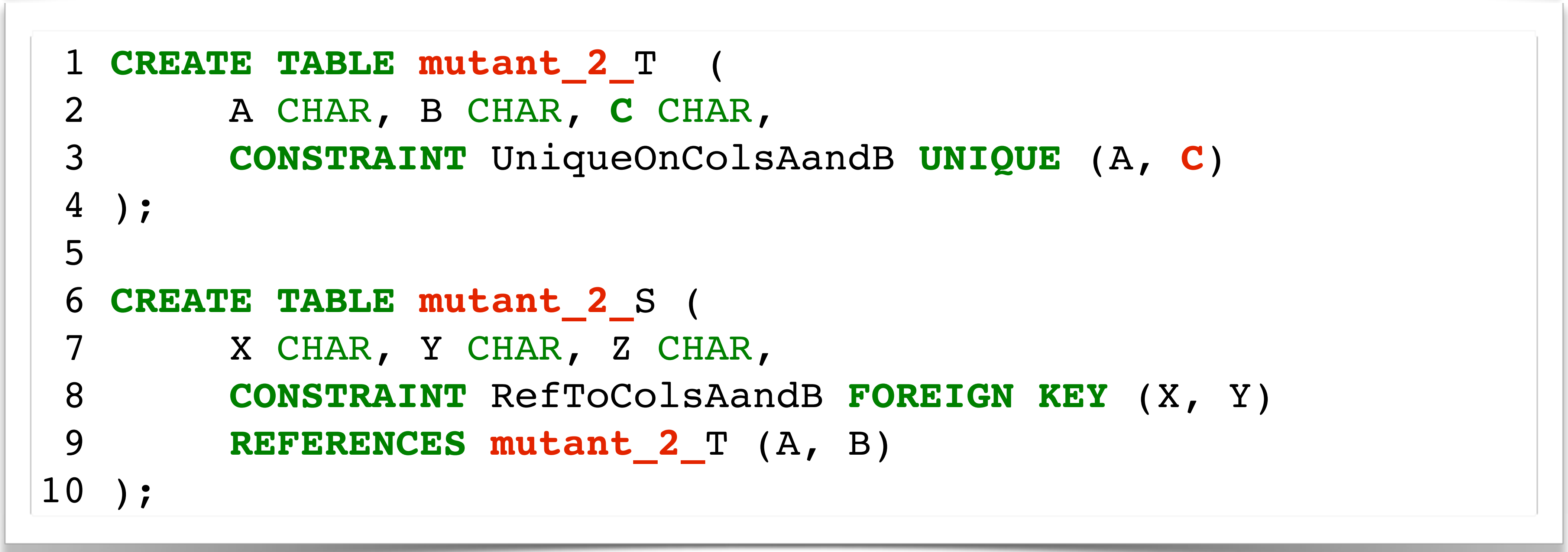


```
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, C)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```

Full Schemata Approach

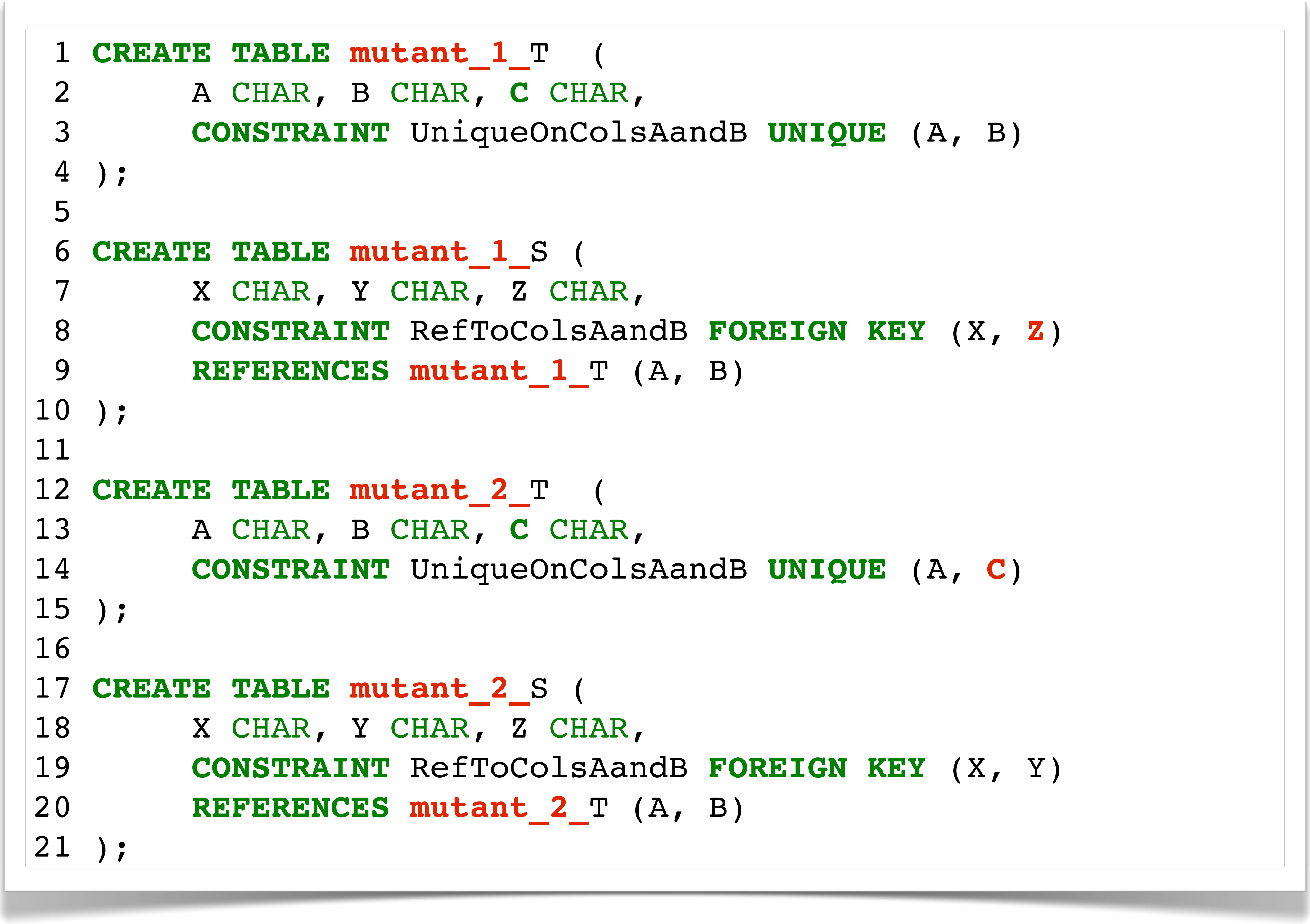


```
CREATE TABLE mutant_1_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE mutant_1_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES mutant_1_T (A, B)
);
```



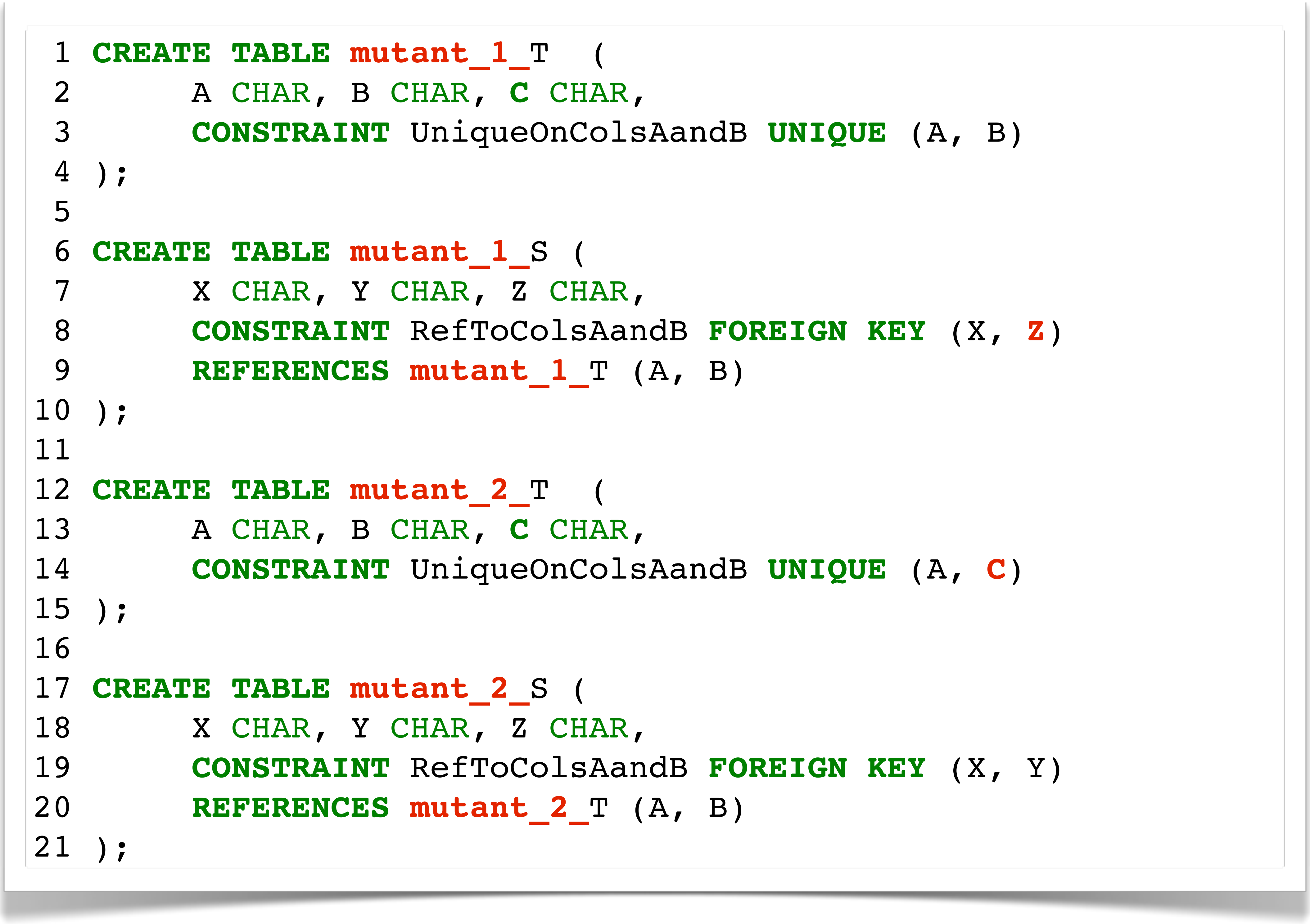
```
CREATE TABLE mutant_2_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, C)
);

CREATE TABLE mutant_2_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES mutant_2_T (A, B)
);
```

Full Schemata Approach



```
CREATE TABLE mutant_1_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE mutant_1_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES mutant_1_T (A, B)
);

CREATE TABLE mutant_2_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, C)
);

CREATE TABLE mutant_2_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES mutant_2_T (A, B)
);
```

Full Schemata Approach

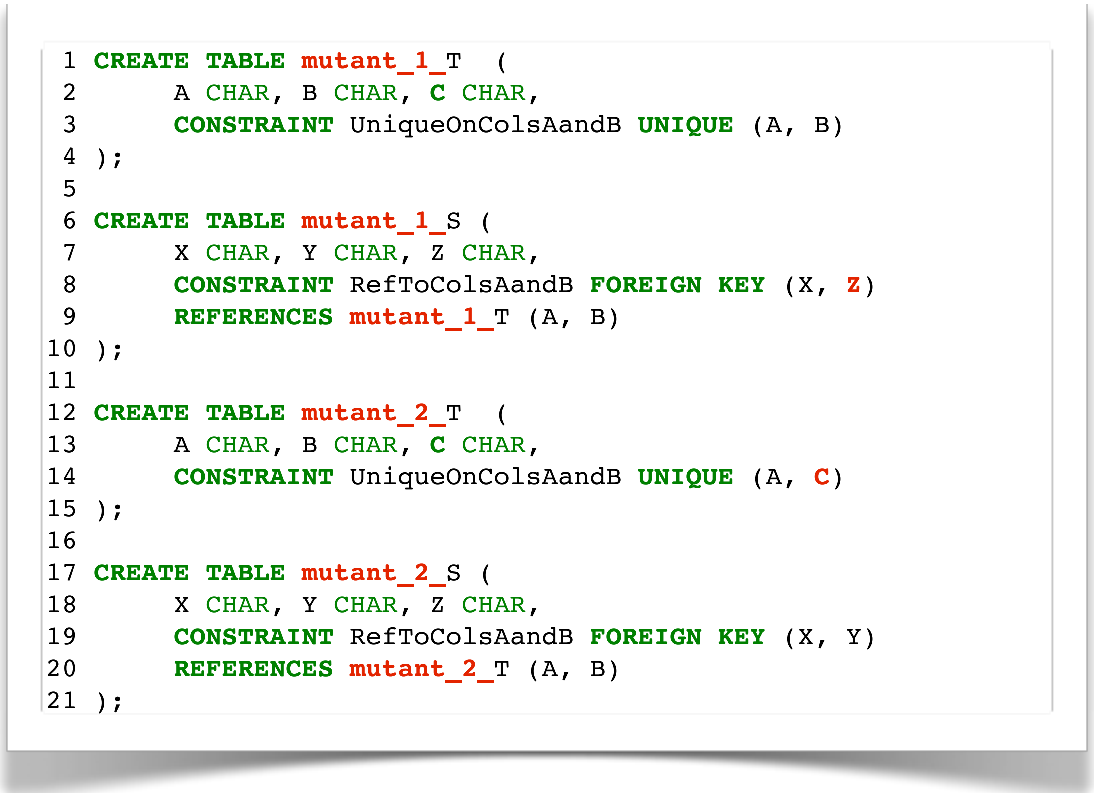


```
CREATE TABLE mutant_1_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE mutant_1_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES mutant_1_T (A, B)
);

CREATE TABLE mutant_2_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, C)
);

CREATE TABLE mutant_2_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES mutant_2_T (A, B)
);
```

Full Schemata Approach

Insert Statements

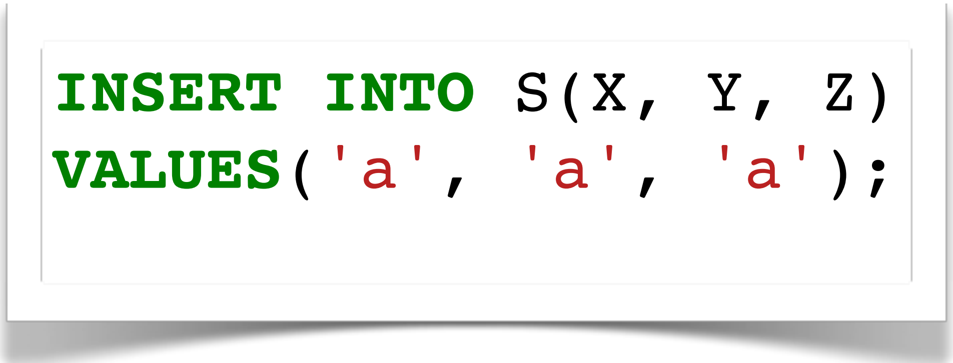
Database (mutated)



```
CREATE TABLE mutant_1_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE mutant_1_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES mutant_1_T (A, B)
);

CREATE TABLE mutant_2_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, C)
);

CREATE TABLE mutant_2_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES mutant_2_T (A, B)
);
```

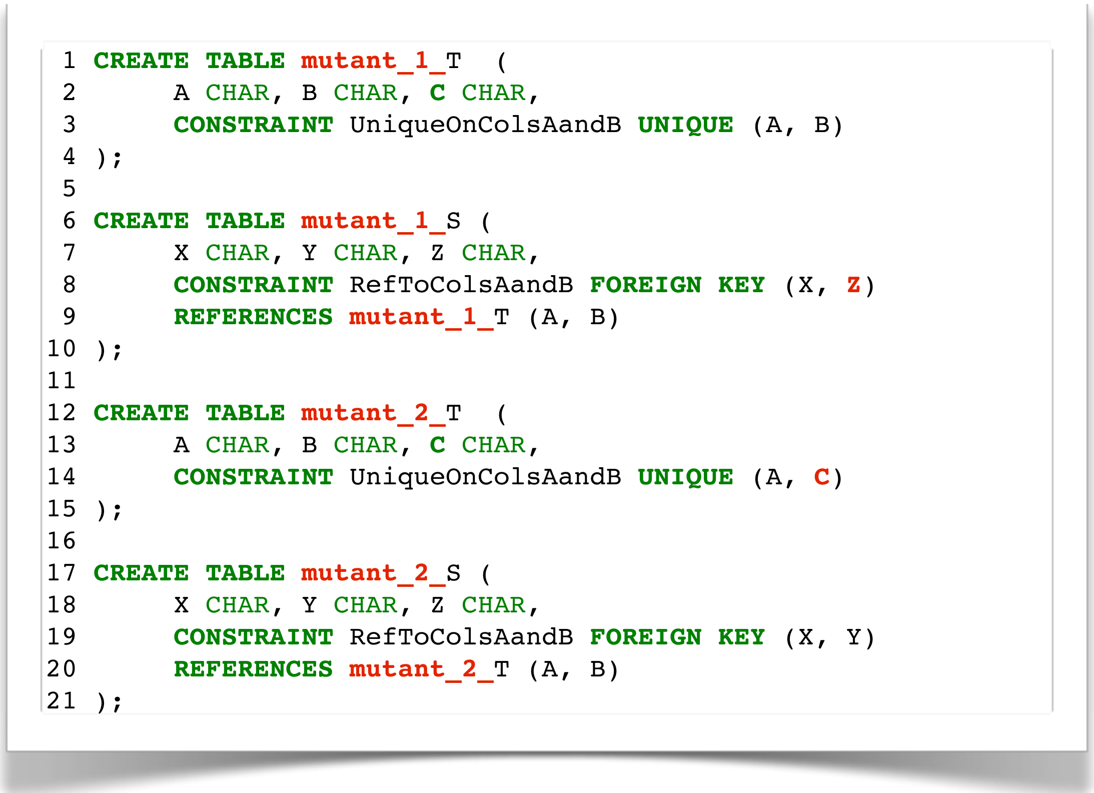
Full Schemata Approach

Insert Statements



```
INSERT INTO S(X, Y, Z)
VALUES('a', 'a', 'a');
```

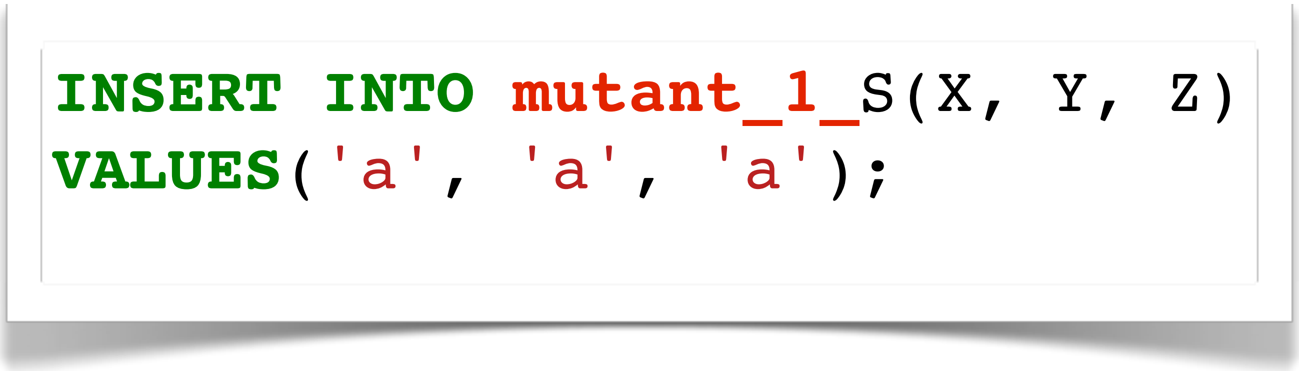
Database (mutated)



```
CREATE TABLE mutant_1_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE mutant_1_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES mutant_1_T (A, B)
);

CREATE TABLE mutant_2_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, C)
);

CREATE TABLE mutant_2_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES mutant_2_T (A, B)
);
```

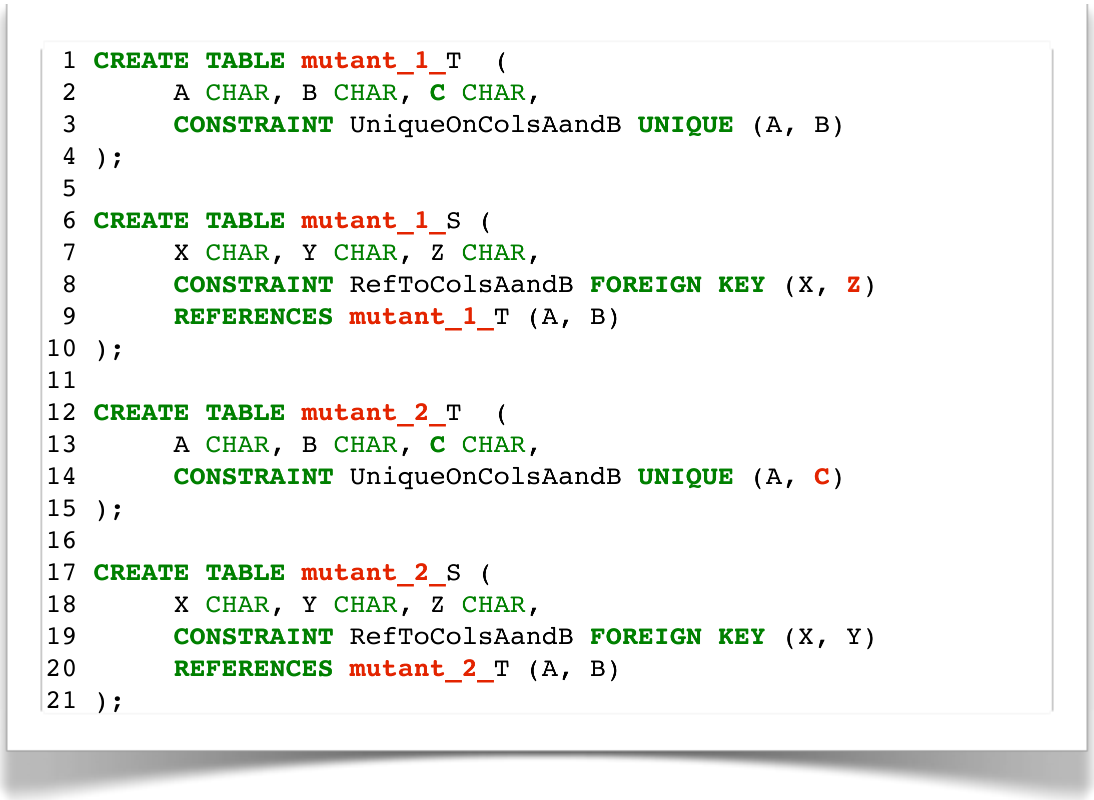
Full Schemata Approach

Insert Statements



```
INSERT INTO mutant_1_S(X, Y, Z)
VALUES('a', 'a', 'a');
```

Database (mutated)

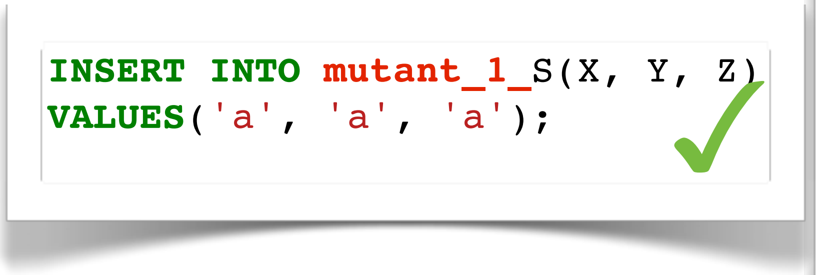


```
CREATE TABLE mutant_1_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE mutant_1_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES mutant_1_T (A, B)
);

CREATE TABLE mutant_2_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, C)
);

CREATE TABLE mutant_2_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES mutant_2_T (A, B)
);
```

Full Schemata Approach

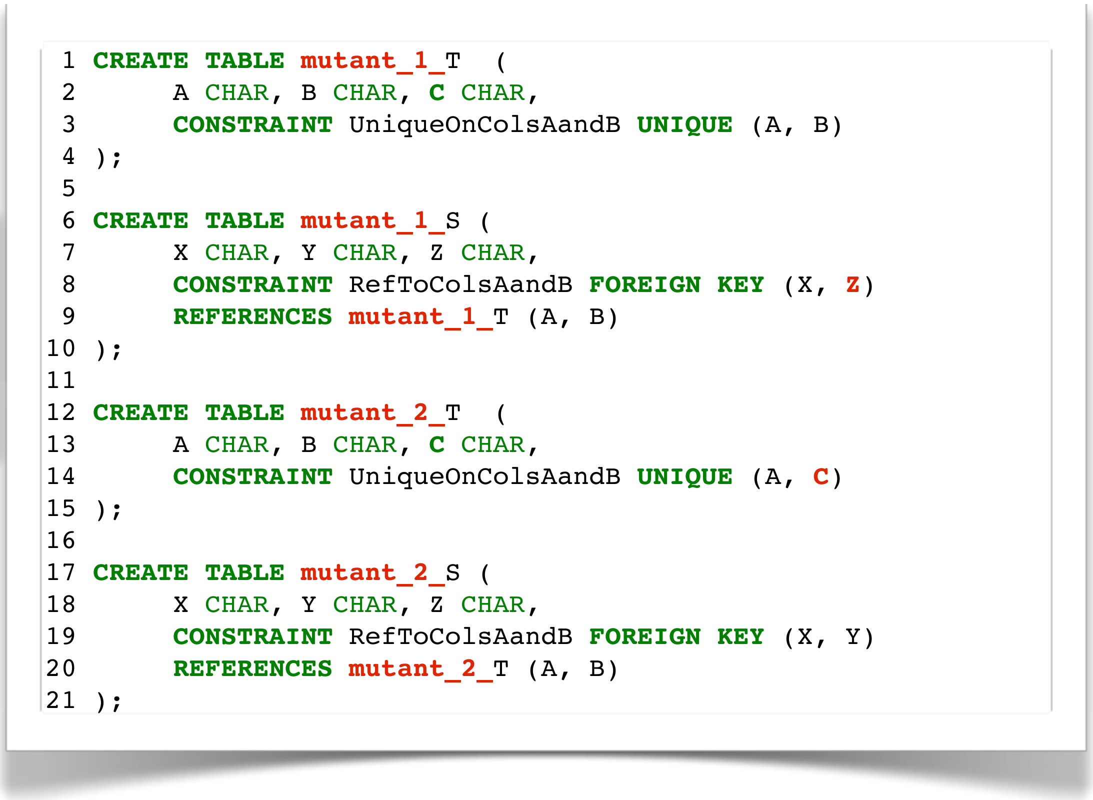
Insert Statements

Database (mutated)



```
INSERT INTO mutant_1_S(X, Y, Z)
VALUES('a', 'a', 'a');
```

✓

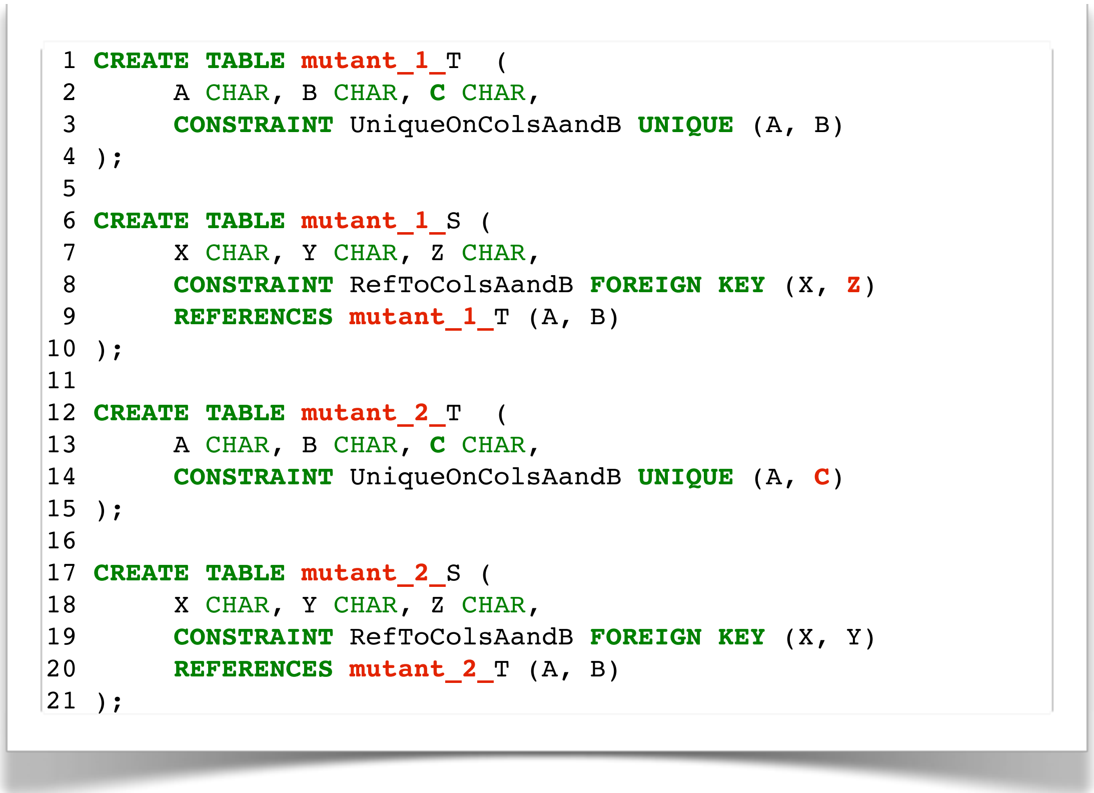


```
CREATE TABLE mutant_1_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE mutant_1_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES mutant_1_T (A, B)
);

CREATE TABLE mutant_2_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, C)
);

CREATE TABLE mutant_2_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES mutant_2_T (A, B)
);
```

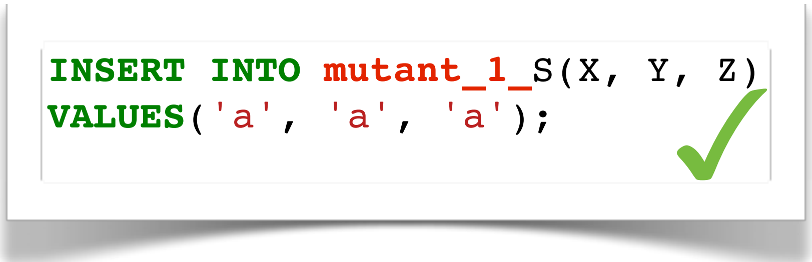
Full Schemata Approach

Insert Statements

Database (mutated)



```
CREATE TABLE mutant_1_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE mutant_1_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES mutant_1_T (A, B)
);

CREATE TABLE mutant_2_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, C)
);

CREATE TABLE mutant_2_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES mutant_2_T (A, B)
);
```



```
INSERT INTO mutant_1_S(X, Y, Z)
VALUES('a', 'a', 'a');
```

✓

Chris J. Wright - c.wright@dcs.shef.ac.uk

Full Schemata Approach

Full Schemata Approach

Create tables once...

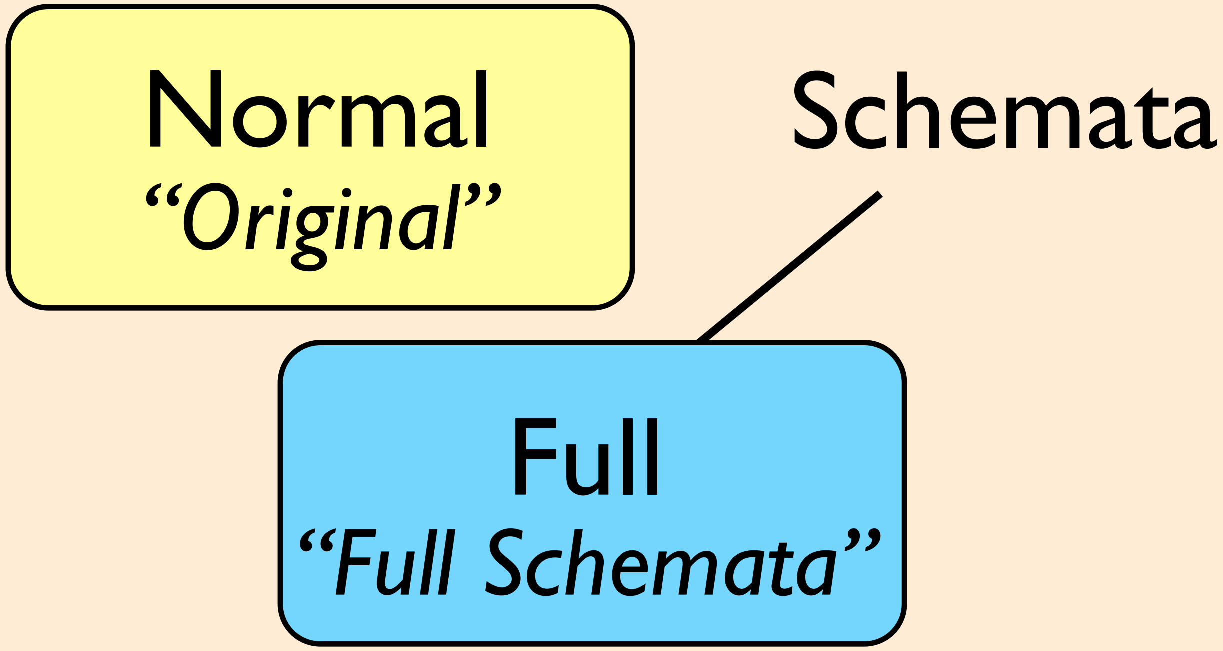
Full Schemata Approach

Create tables once...

...rewrite queries to match

Mutation Analysis Approaches

Mutant Representation



Normal

“Original”

Schemata

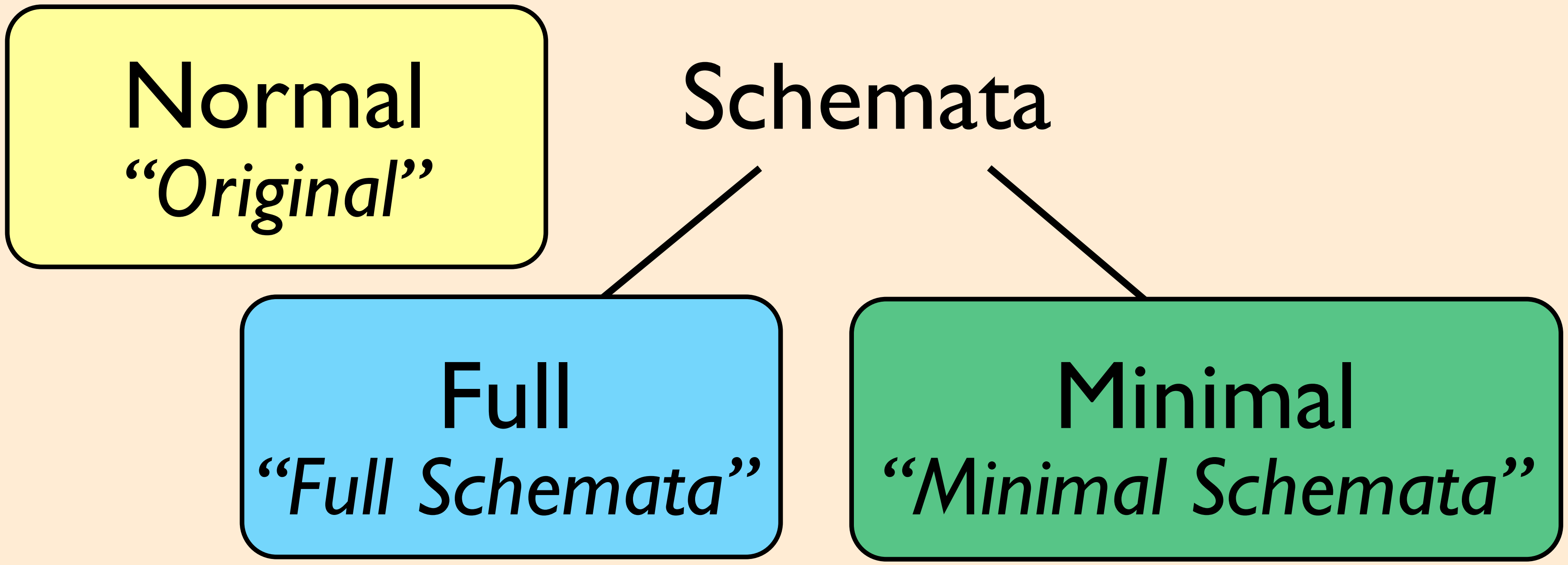
Full

“Full Schemata”

Parallelisation Strategy

Mutation Analysis Approaches

Mutant Representation



Normal

"Original"

Schemata

Full

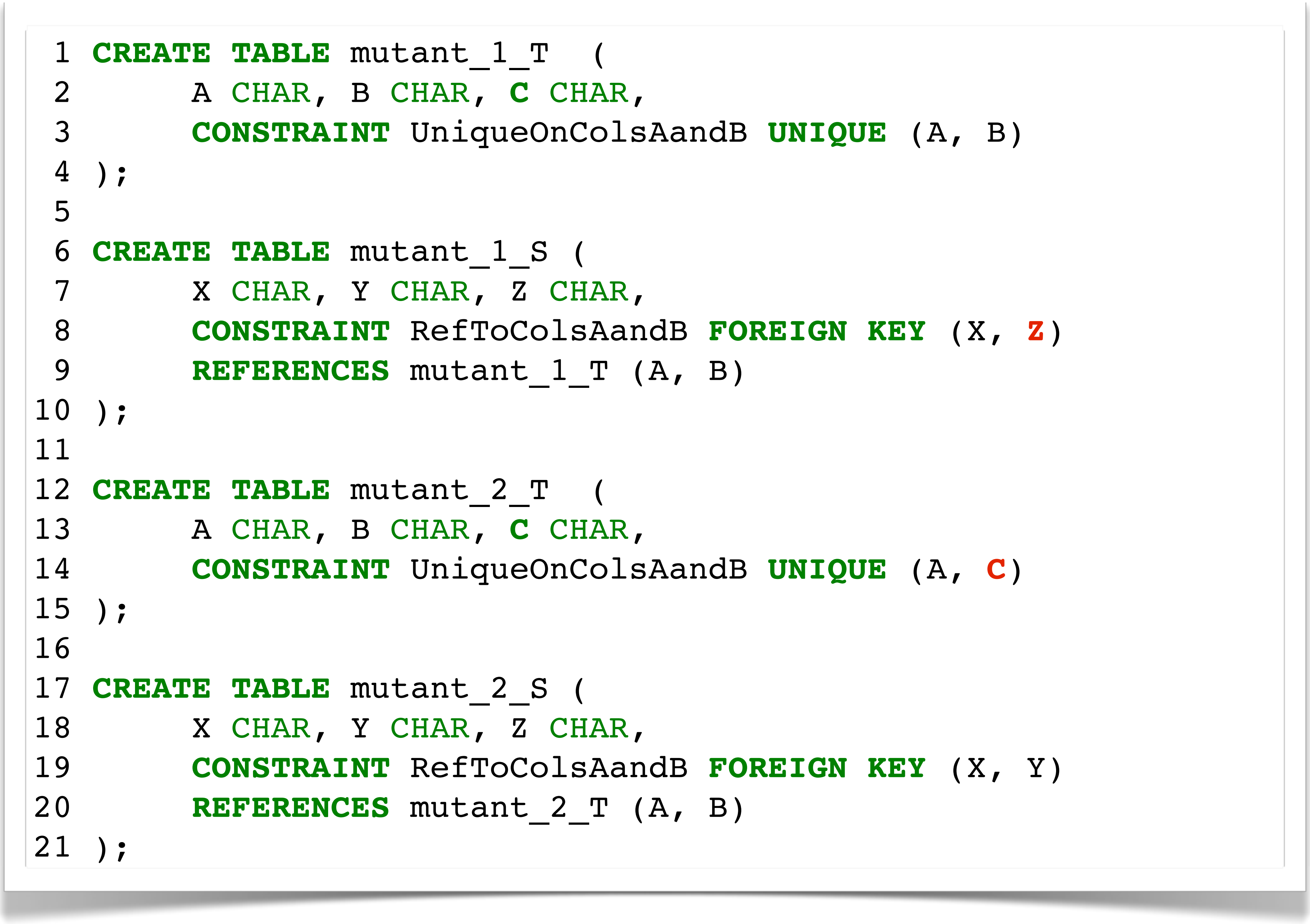
"Full Schemata"

Minimal

"Minimal Schemata"

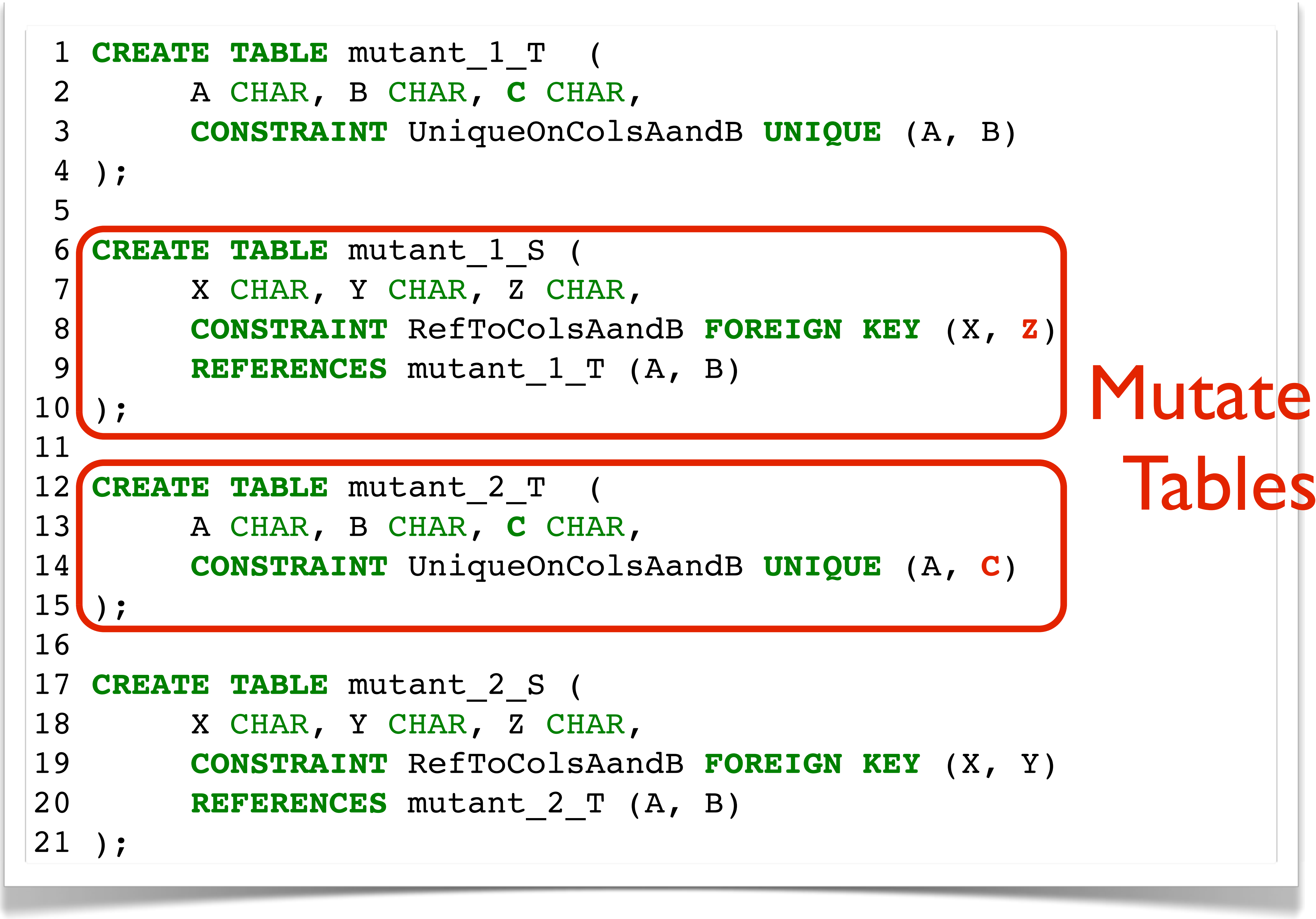
Parallelisation Strategy

Full Schemata Approach



```
CREATE TABLE mutant_1_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE mutant_1_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES mutant_1_T (A, B)
);

CREATE TABLE mutant_2_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, C)
);

CREATE TABLE mutant_2_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES mutant_2_T (A, B)
);
```

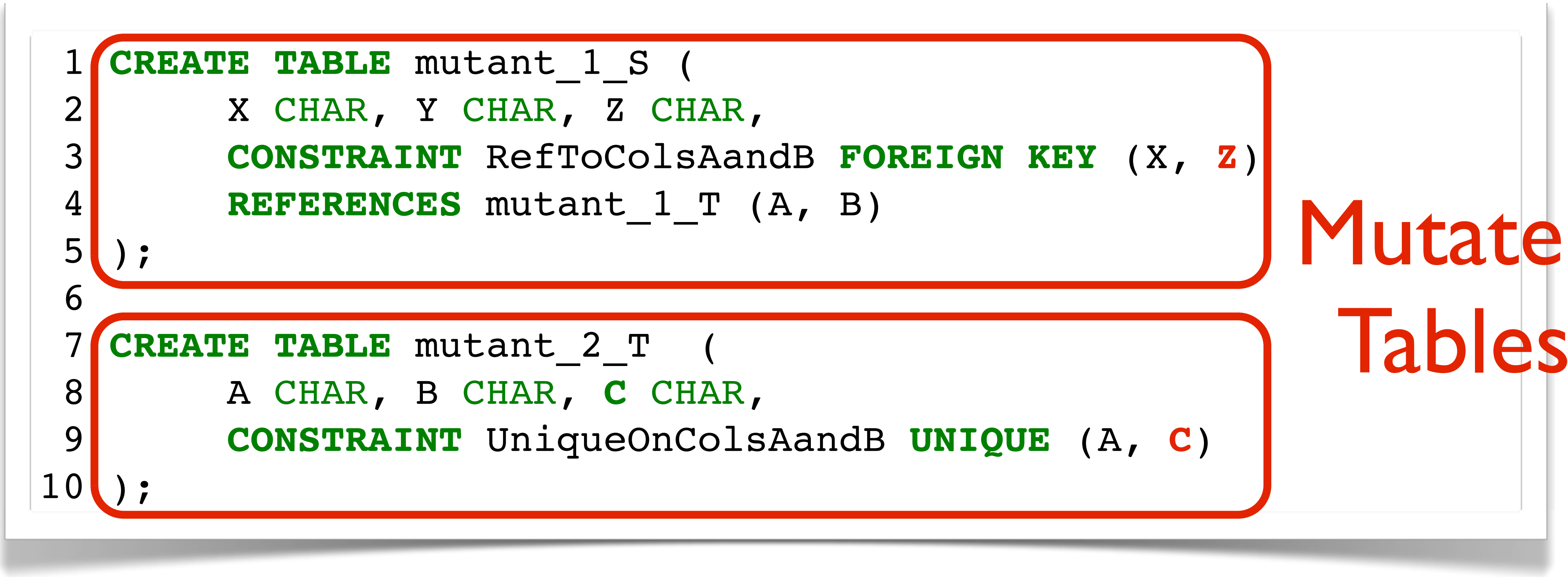
Full Schemata Approach



```
CREATE TABLE mutant_1_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE mutant_1_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES mutant_1_T (A, B)
);

CREATE TABLE mutant_2_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, C)
);

CREATE TABLE mutant_2_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES mutant_2_T (A, B)
);
```

Mutated Tables

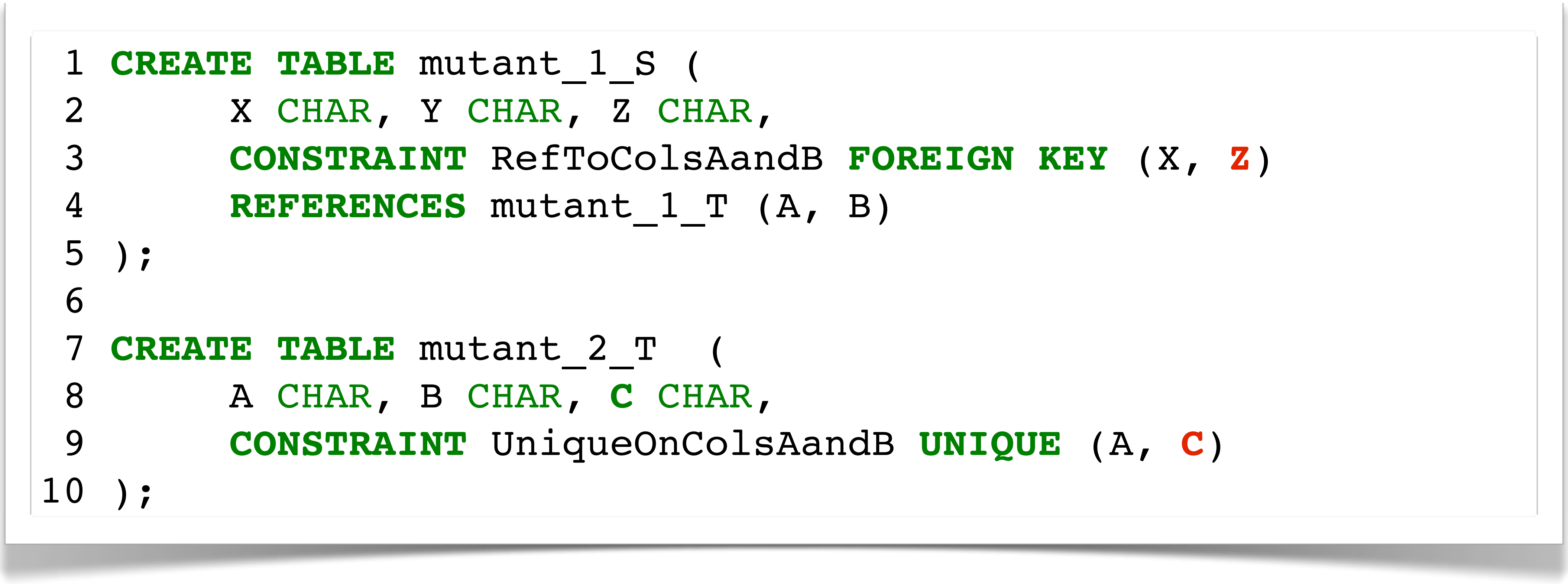
Minimal Schemata Approach



```
CREATE TABLE mutant_1_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES mutant_1_T (A, B)
);

CREATE TABLE mutant_2_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, C)
);
```

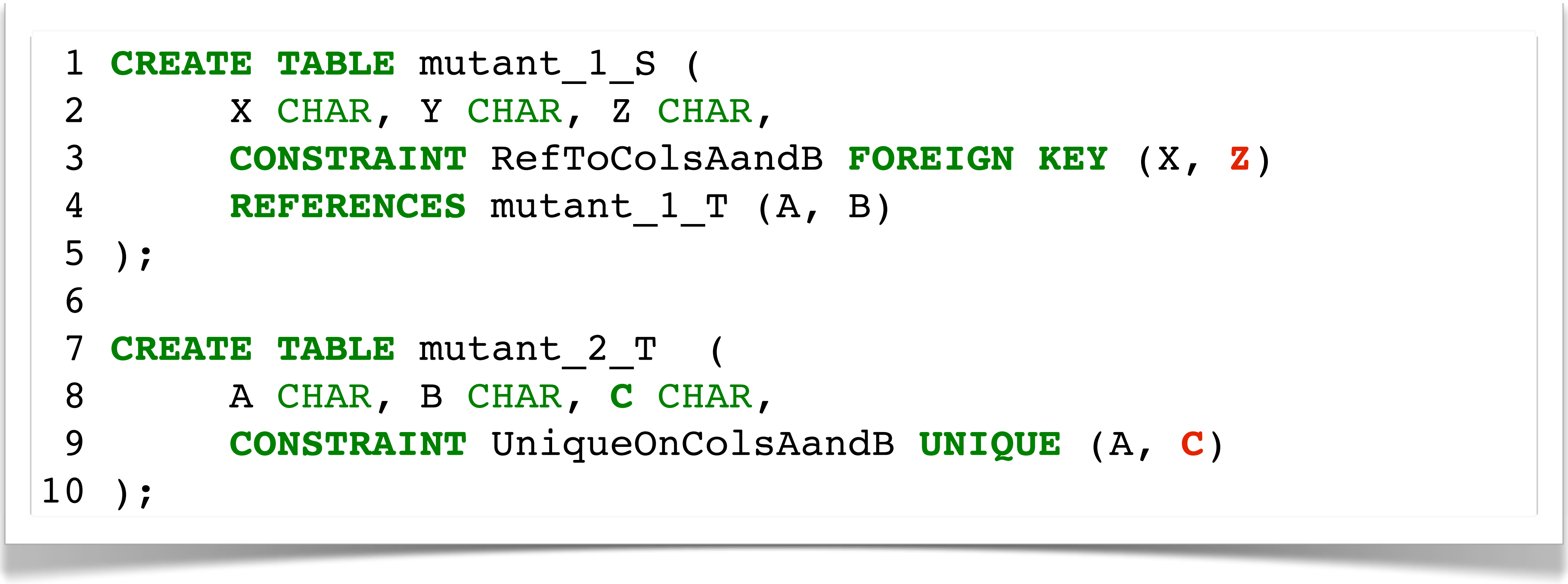
Mutated Tables

Minimal Schemata Approach



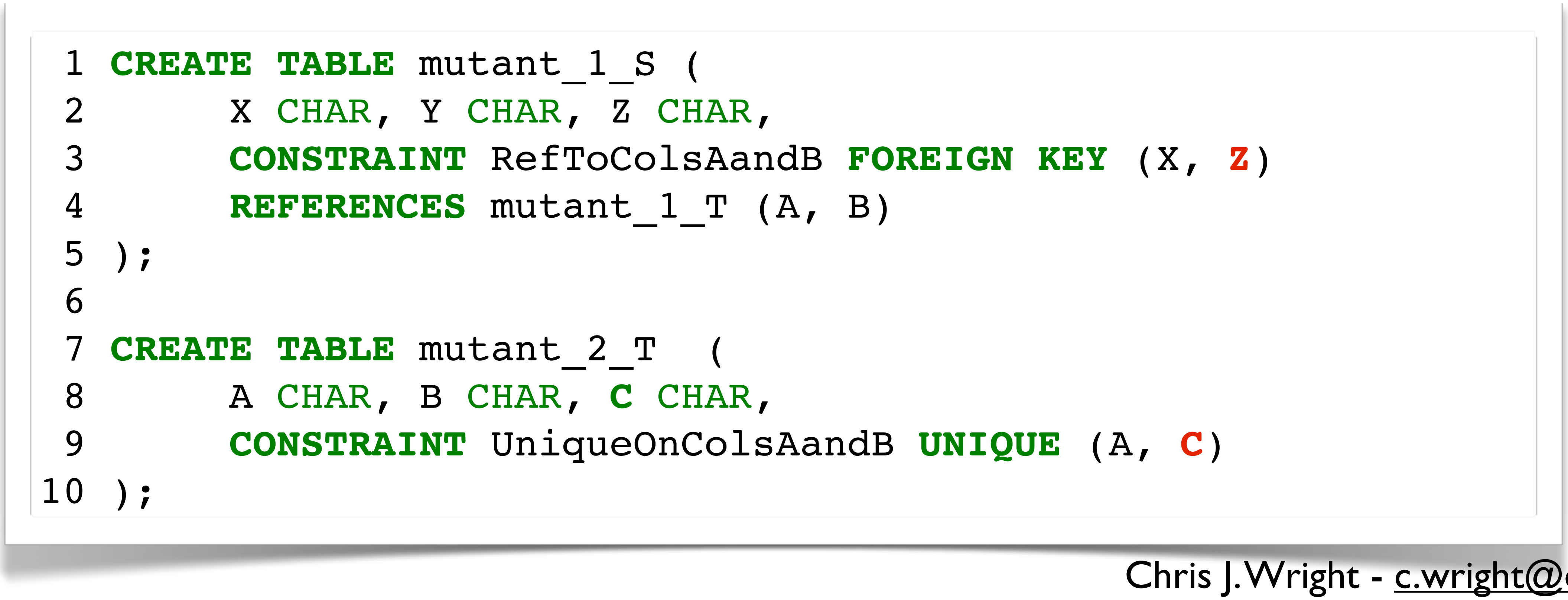
```sql
CREATE TABLE mutant_1_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES mutant_1_T (A, B)
);

CREATE TABLE mutant_2_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, C)
);
```

Minimal Schemata Approach



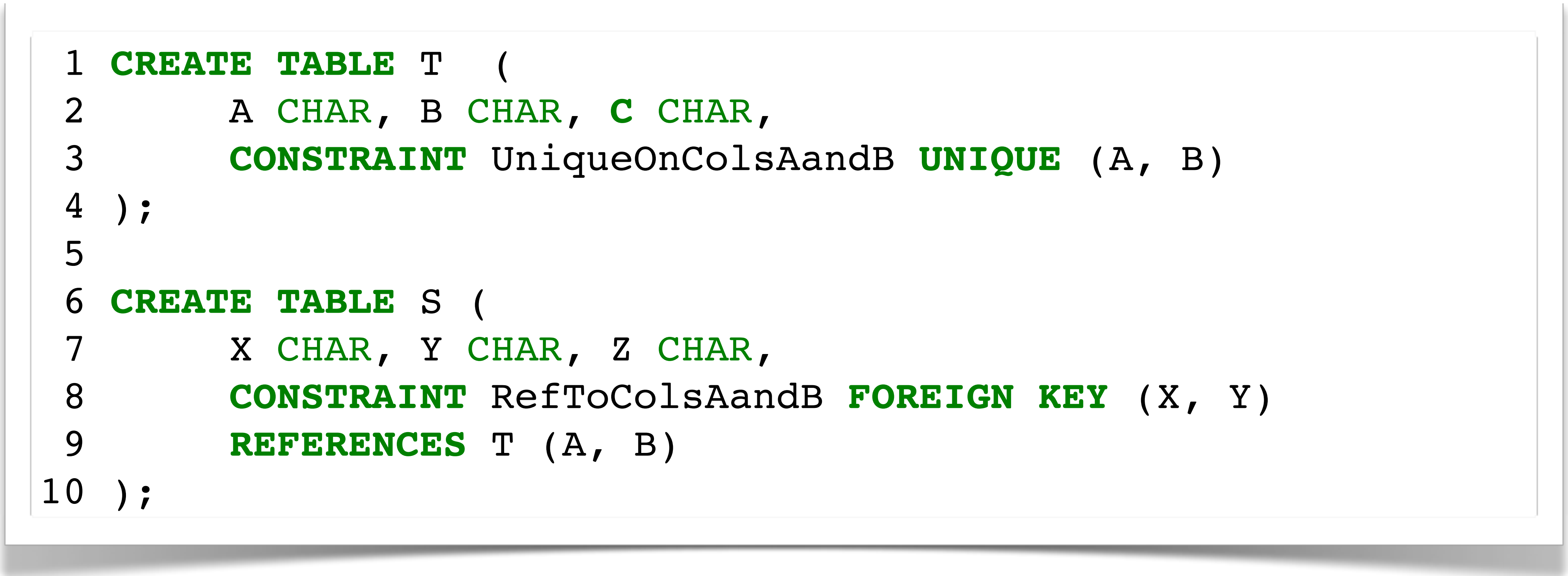
```
CREATE TABLE mutant_1_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES mutant_1_T (A, B)
);

CREATE TABLE mutant_2_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, C)
);
```

Minimal Schemata Approach

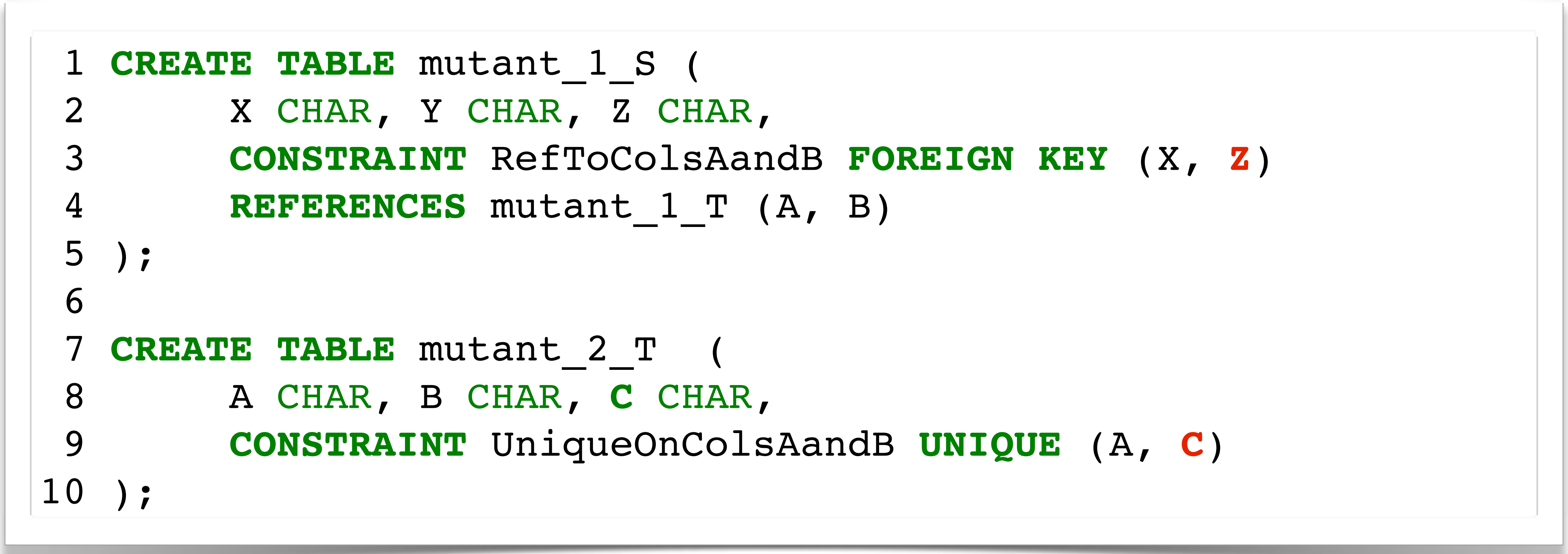


```sql
CREATE TABLE mutant_1_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES mutant_1_T (A, B)
);

CREATE TABLE mutant_2_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, C)
);
```

Minimal Schemata Approach

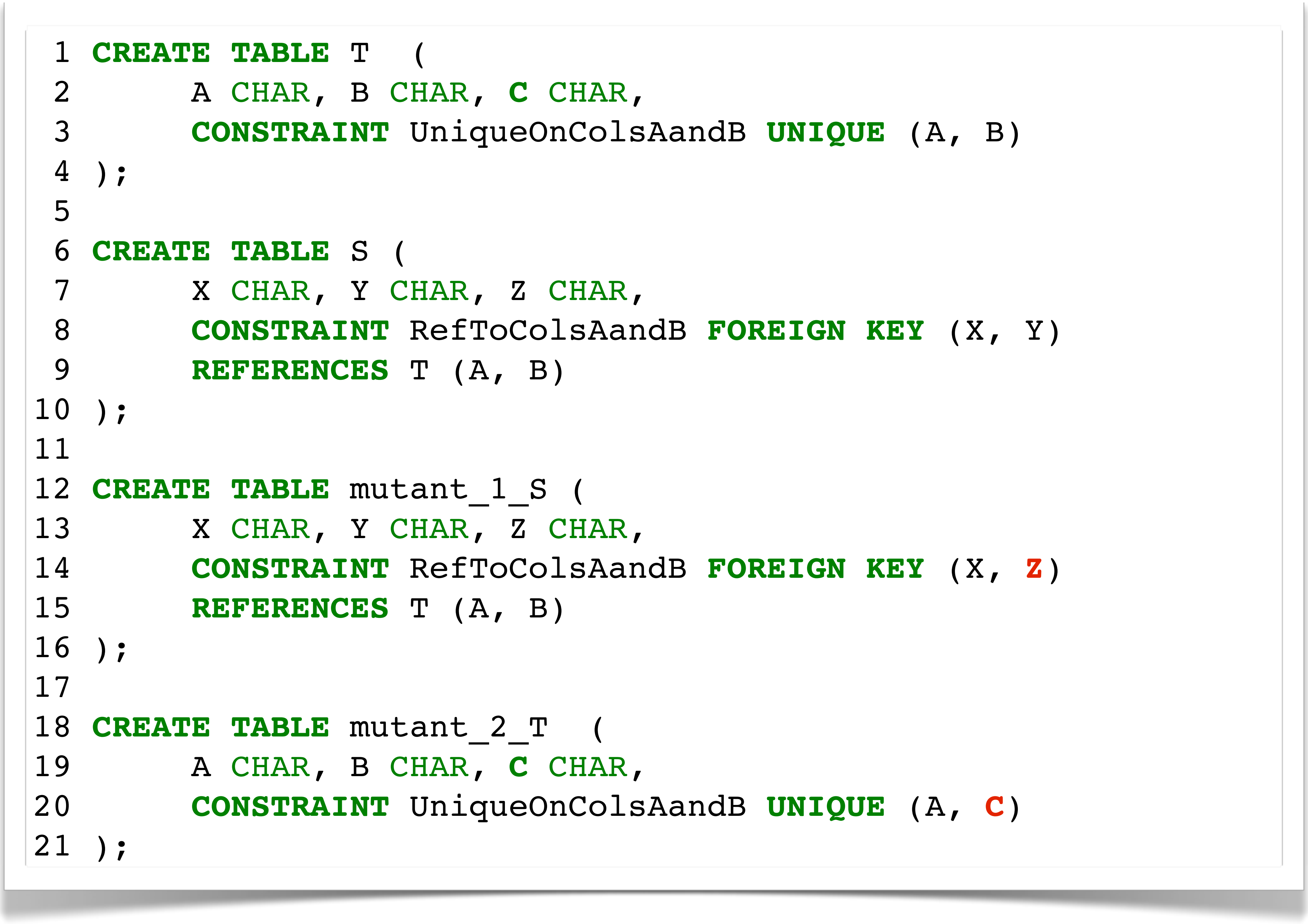


```sql
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);
```



```sql
CREATE TABLE mutant_1_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES mutant_1_T (A, B)
);

CREATE TABLE mutant_2_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, C)
);
```

Minimal Schemata Approach



```sql
CREATE TABLE T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, B)
);

CREATE TABLE S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Y)
     REFERENCES T (A, B)
);

CREATE TABLE mutant_1_S (
     X CHAR, Y CHAR, Z CHAR,
     CONSTRAINT RefToColsAandB FOREIGN KEY (X, Z)
     REFERENCES T (A, B)
);

CREATE TABLE mutant_2_T  (
     A CHAR, B CHAR, C CHAR,
     CONSTRAINT UniqueOnColsAandB UNIQUE (A, C)
);
```

Minimal Schemata Approach

Minimal Schemata Approach

Create tables once...

Minimal Schemata Approach

Create tables once...

...only mutated tables...

Minimal Schemata Approach

Create tables once...

...only mutated tables...

...reduce queries executed

Minimal Schemata Approach

Create tables once...

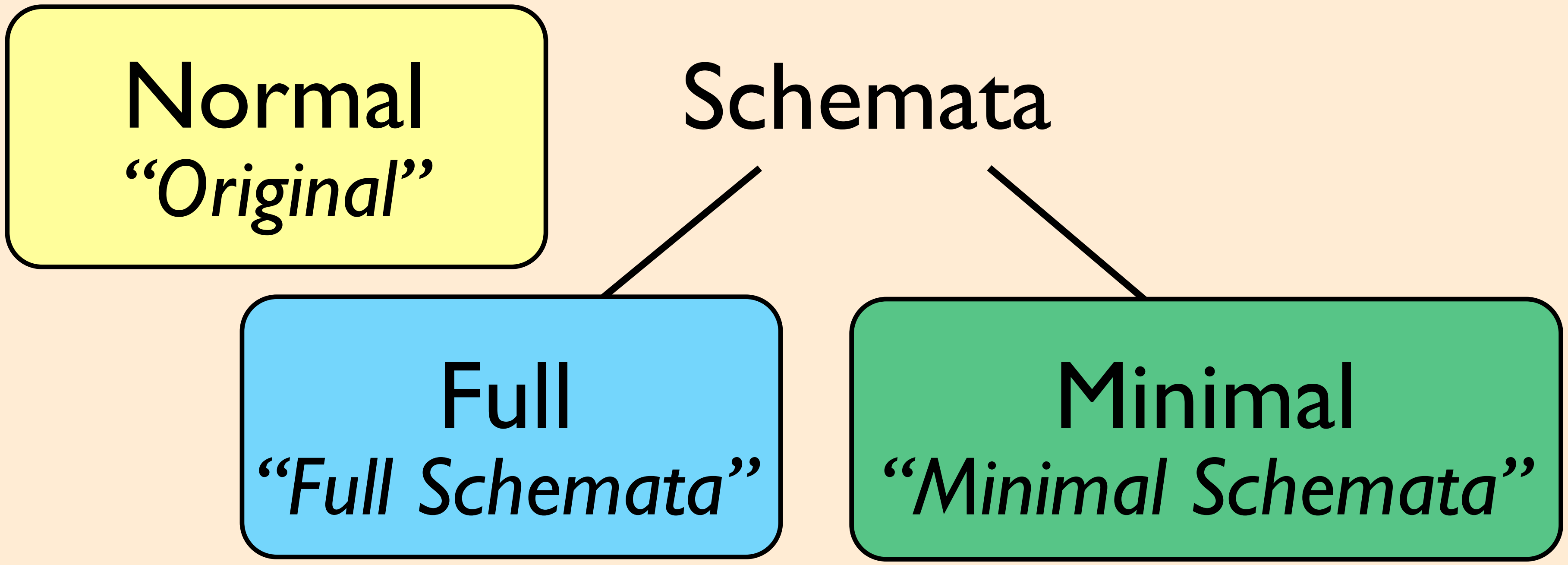
Plus one copy for foreign keys

...only mutated tables...

...reduce queries executed

Mutation Analysis Approaches

Mutant Representation



Normal
“Original”

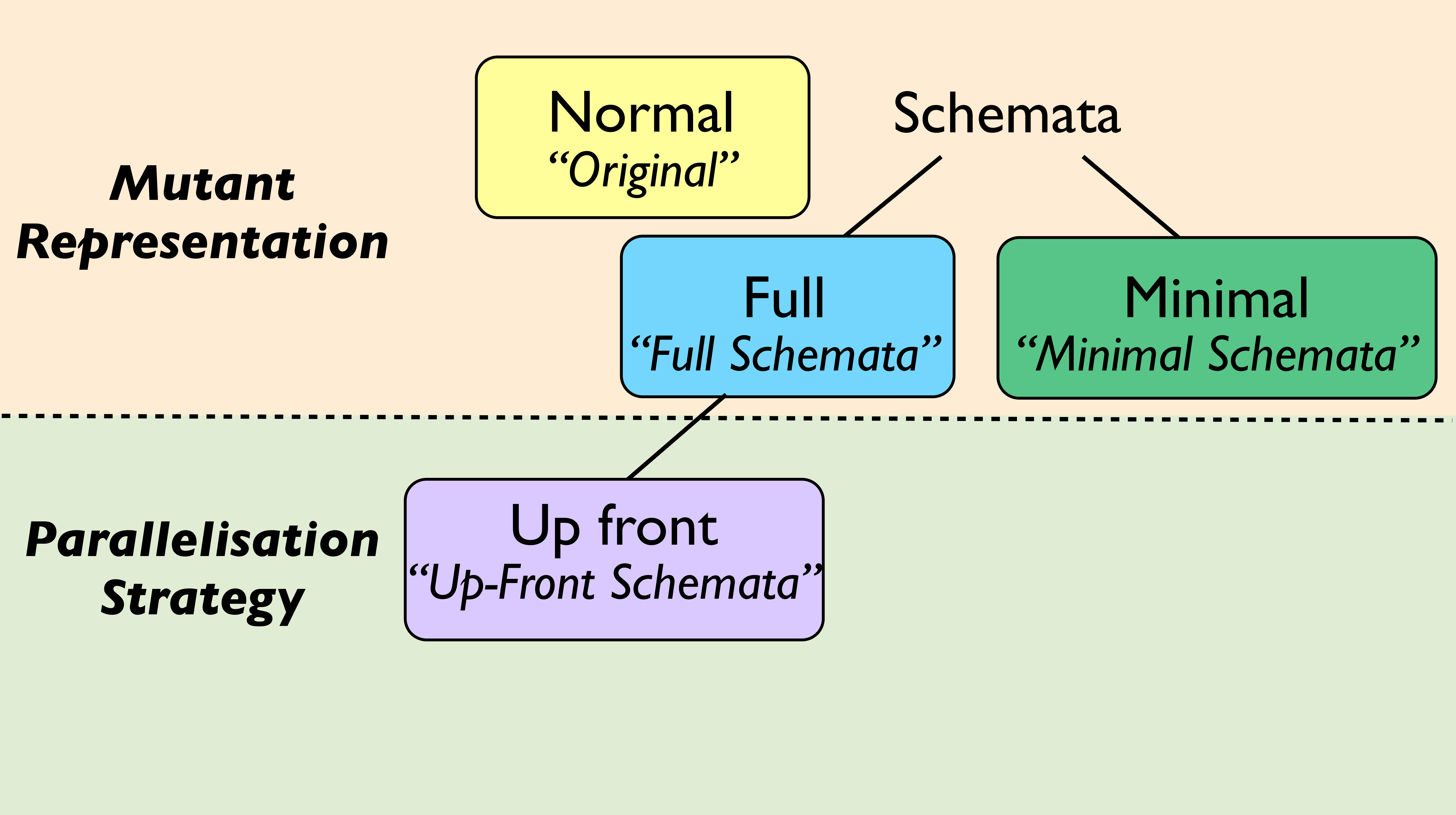
Schemata

Full
“Full Schemata”

Minimal
“Minimal Schemata”

Parallelisation Strategy

Mutation Analysis Approaches



Mutant Representation

Normal
"Original"

Schemata

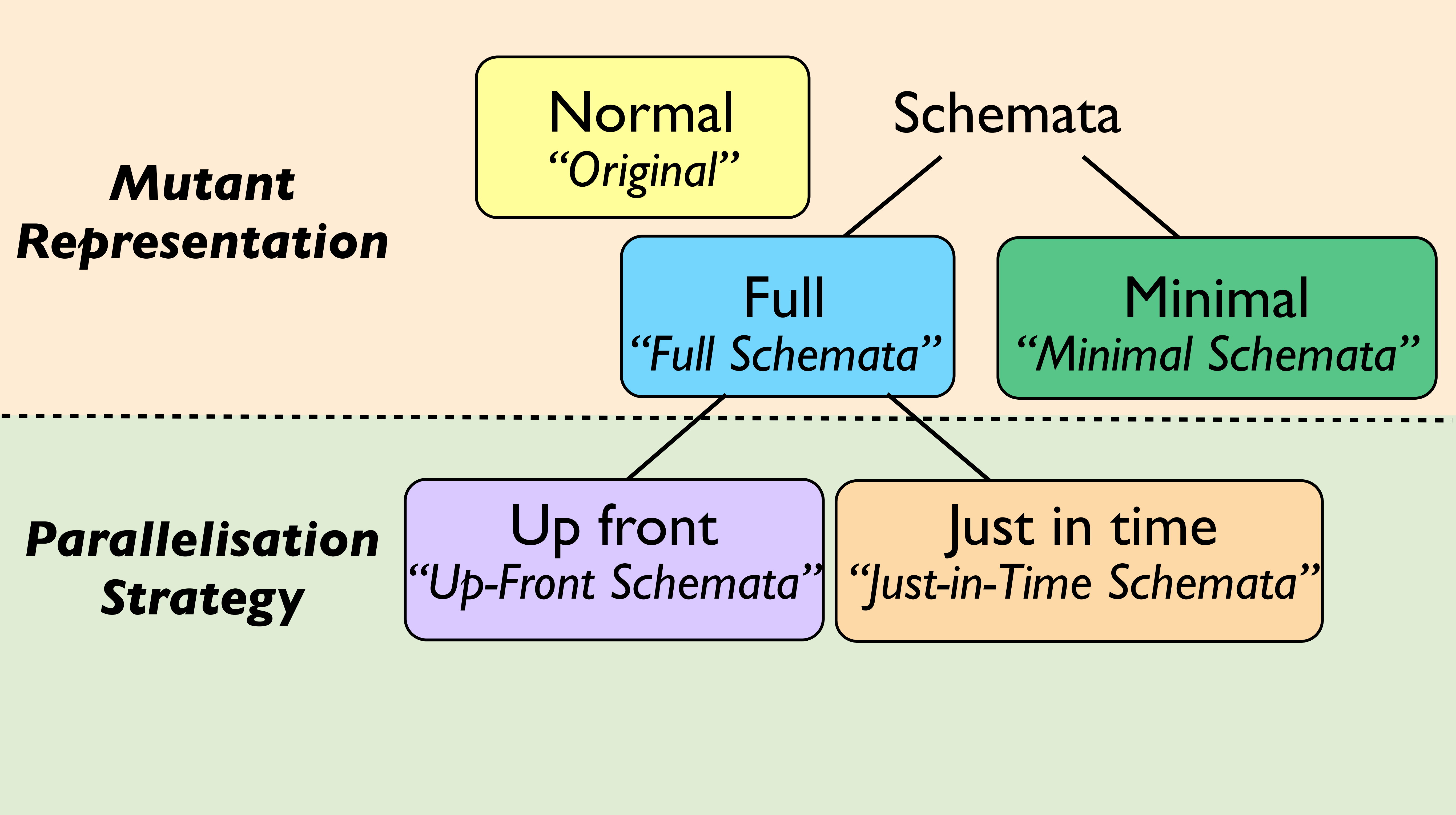
Full
"Full Schemata"

Minimal
"Minimal Schemata"

Parallelisation Strategy

Up front
"Up-Front Schemata"

Mutation Analysis Approaches



Mutant Representation

Normal
"Original"

Schemata

Full
"Full Schemata"

Minimal
"Minimal Schemata"

Parallelisation Strategy

Up front
"Up-Front Schemata"

Just in time
"Just-in-Time Schemata"

Parallelisation

‘Just-in-Time Schemata’

Parallelisation

‘Just-in-Time Schemata’

Make the ‘Original’ approach parallel

Parallelisation

‘Just-in-Time Schemata’

Make the ‘Original’ approach parallel

‘Up-Front Schemata’

Parallelisation

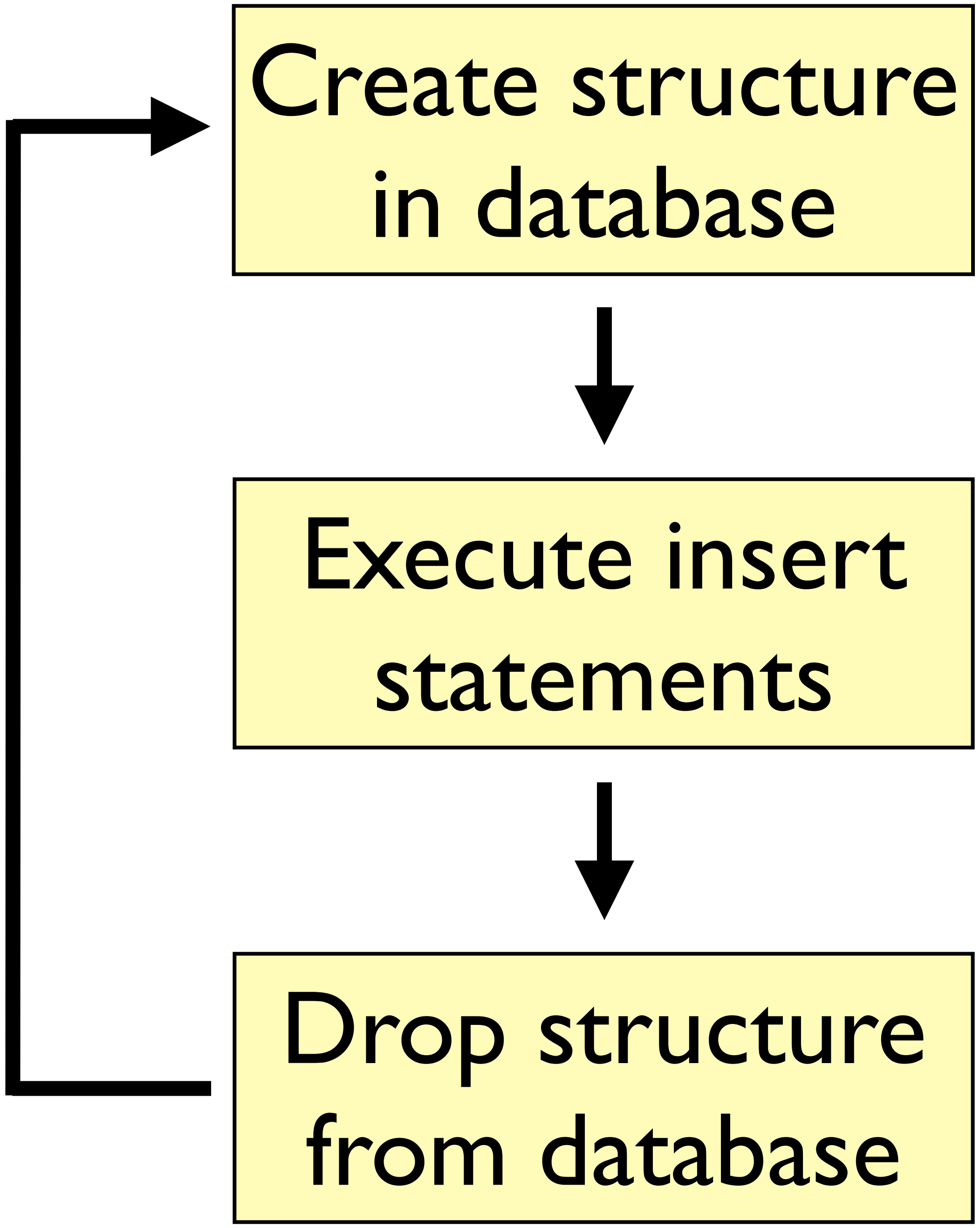
‘Just-in-Time Schemata’

Make the ‘Original’ approach parallel

‘Up-Front Schemata’

Make the ‘Full Schemata’ approach parallel

Original Approach

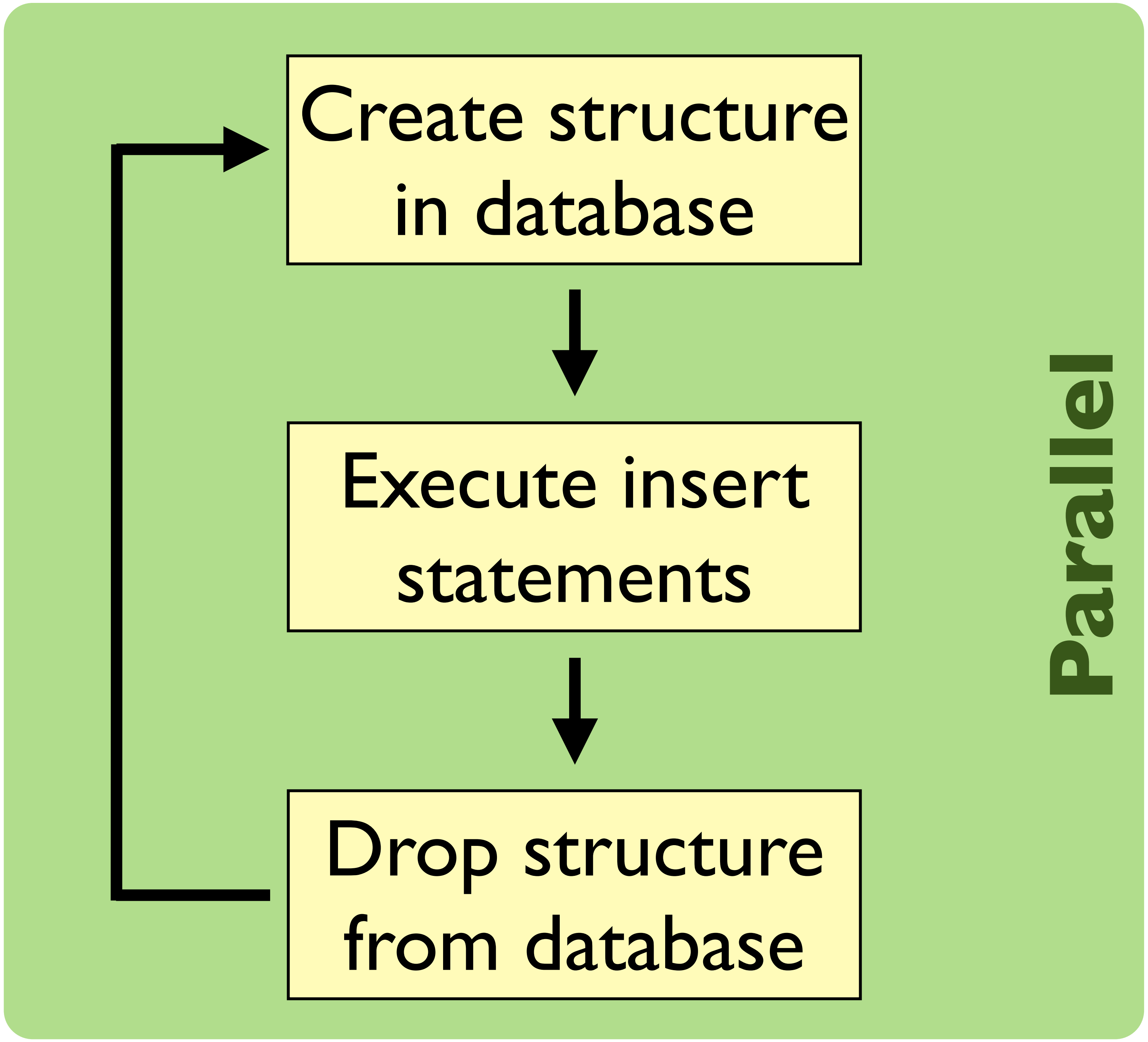


Create structure in database

Execute insert statements

Drop structure from database

'Just-in-Time' Approach

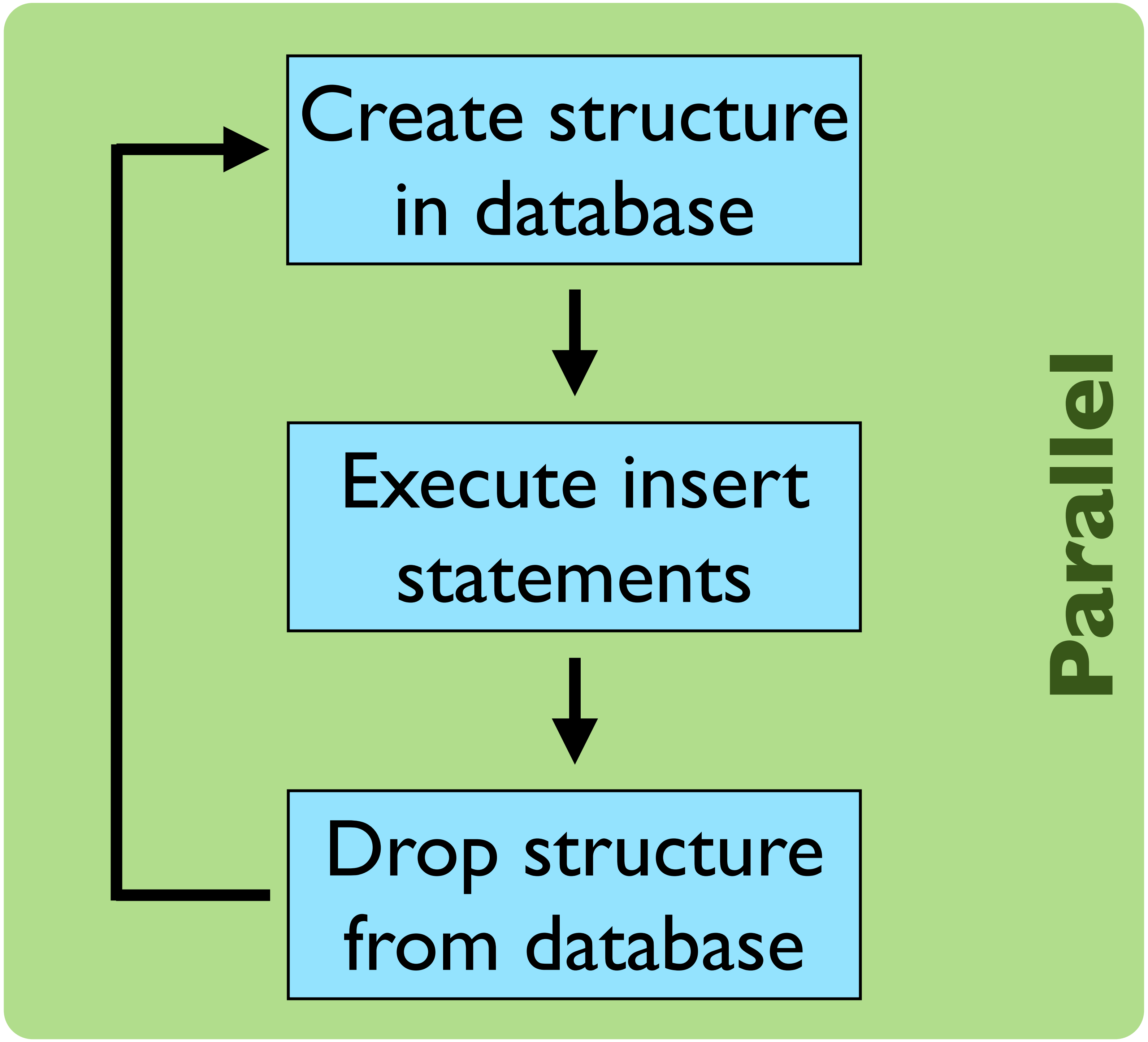


Create structure in database

Execute insert statements

Drop structure from database

'Just-in-Time' Approach

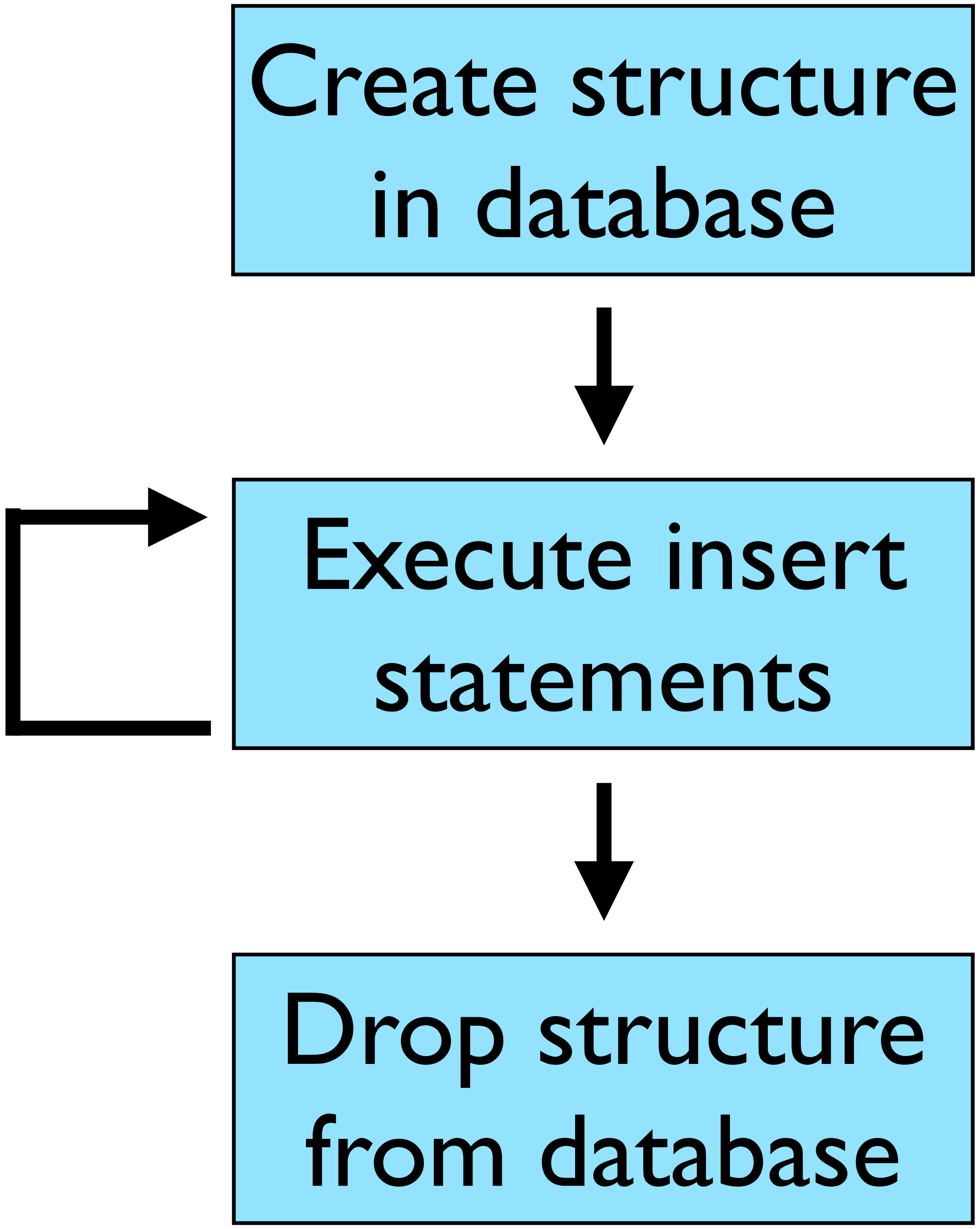


Create structure in database

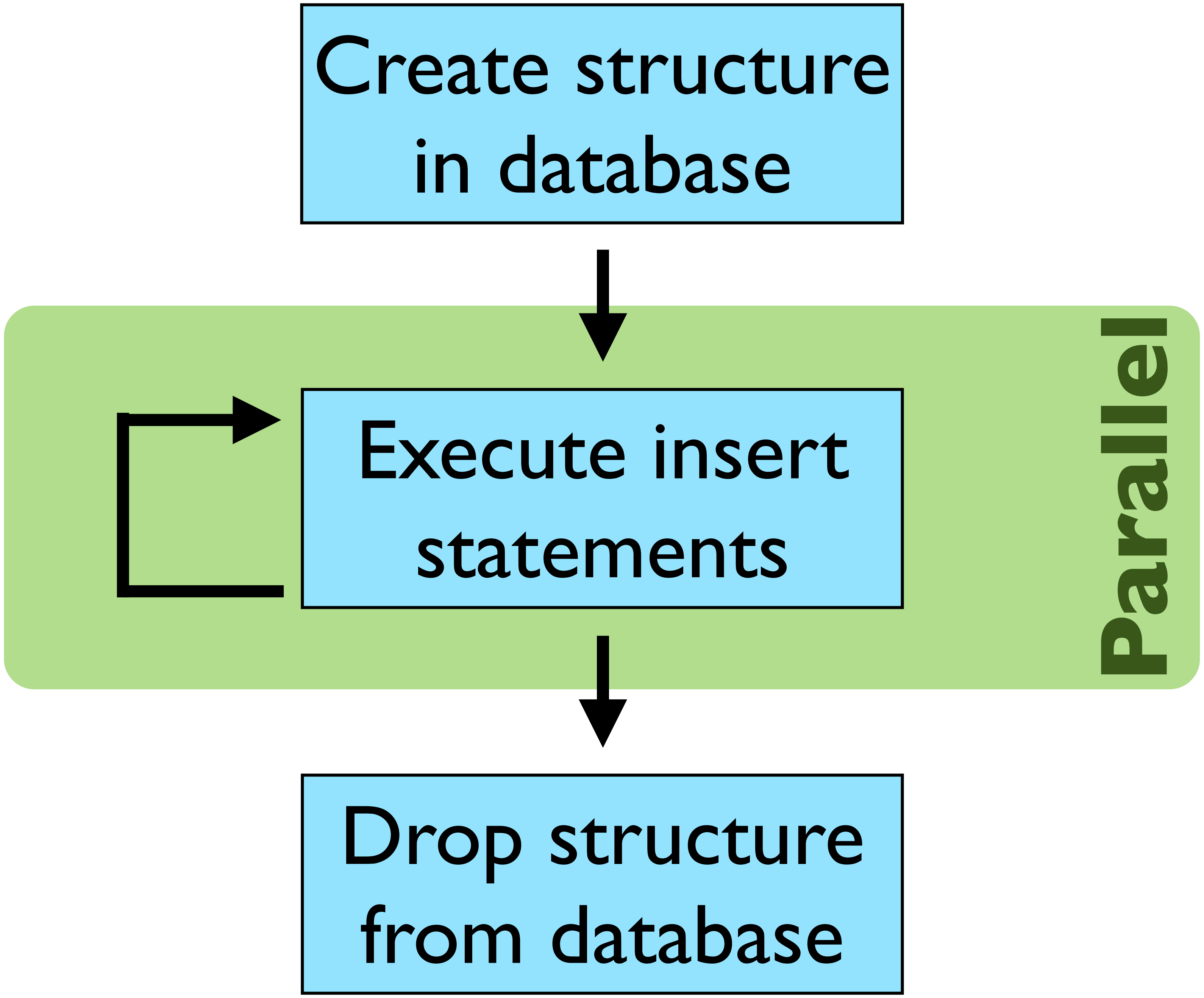
Execute insert statements

Drop structure from database

Mutant Schemata Approach


Create structure in database
Execute insert statements
Drop structure from database

‘Up-Front’ Approach



Create structure in database

Execute insert statements

Drop structure from database

Empirical Study

Empirical Study

Evaluation Metric	*Mutation Time*

Empirical Study

Evaluation Metric	*Mutation Time*
Approaches	*5*

Empirical Study

Evaluation Metric	*Mutation Time*
Approaches	5
Case Studies	6

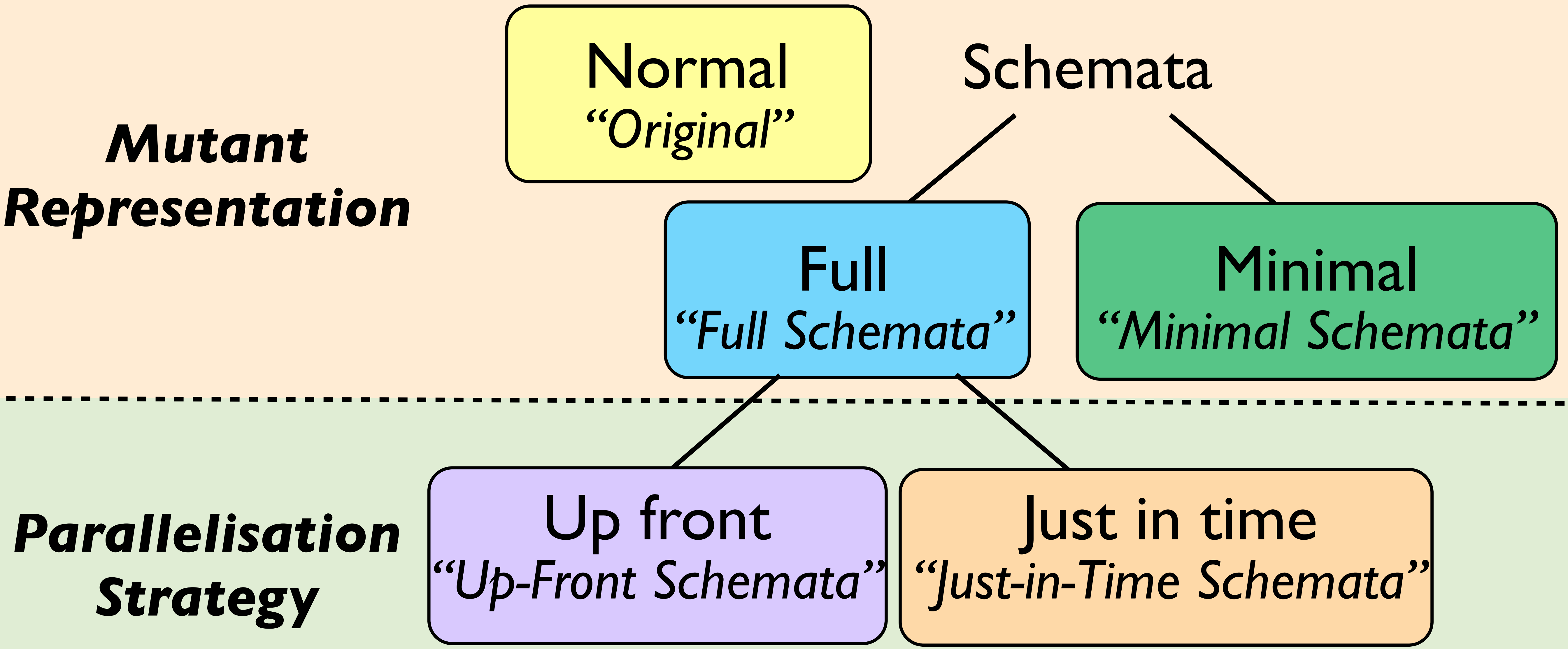
Empirical Study

Evaluation Metric	*Mutation Time*
Approaches	5
Case Studies	6
DBMSs	2

Empirical Study

Evaluation Metric	*Mutation Time*
Approaches	*5*
Case Studies	*6*
DBMSs	*2*
Repetitions	*30*

Empirical Study: Approaches



Mutant Representation

Normal
"Original"

Schemata

Full
"Full Schemata"

Minimal
"Minimal Schemata"

Parallelisation Strategy

Up front
"Up-Front Schemata"

Just in time
"Just-in-Time Schemata"

Empirical Study: Case Studies

Case Study	Tables	Columns	Primary Keys	Foreign Keys	Unique Constraints

Empirical Study: Case Studies

Case Study	Tables	Columns	Primary Keys	Foreign Keys	Unique Constraints
Cloc	*2*	*10*	*0*	*0*	*0*

Empirical Study: Case Studies

Case Study	Tables	Columns	Primary Keys	Foreign Keys	Unique Constraints
Cloc	2	10	0	0	0
JWhoisServer	6	49	6	0	0

Empirical Study: Case Studies

Case Study	Tables	Columns	Primary Keys	Foreign Keys	Unique Constraints
Cloc	2	10	0	0	0
JWhoisServer	6	49	6	0	0
NistDML182	2	32	1	1	0

Empirical Study: Case Studies

Case Study	Tables	Columns	Primary Keys	Foreign Keys	Unique Constraints
Cloc	2	10	0	0	0
JWhoisServer	6	49	6	0	0
NistDML182	2	32	1	1	0
NistDML183	2	6	0	1	1

Empirical Study: Case Studies

Case Study	Tables	Columns	Primary Keys	Foreign Keys	Unique Constraints
Cloc	2	10	0	0	0
JWhoisServer	6	49	6	0	0
NistDML182	2	32	1	1	0
NistDML183	2	6	0	1	1
RiskIt	13	56	11	10	0

Empirical Study: Case Studies

Case Study	Tables	Columns	Primary Keys	Foreign Keys	Unique Constraints
Cloc	2	10	0	0	0
JWhoisServer	6	49	6	0	0
NistDML182	2	32	1	1	0
NistDML183	2	6	0	1	1
RiskIt	13	56	11	10	0
UnixUsage	8	32	7	7	0

Empirical Study: Case Studies

Case Study	Total Constraints	Total Mutants
Cloc	0	30
JWhoisServer	50	184
NistDML182	2	66
NistDML183	2	18
RiskIt	36	160
UnixUsage	23	69

Empirical Study: DBMSs

Empirical Study: DBMSs

PostgreSQL

Empirical Study: DBMSs

PostgreSQL

SQLite

Empirical Study: DBMSs

PostgreSQL

Client-Server Model

SQLite

Empirical Study: DBMSs

PostgreSQL

Client-Server Model

SQLite

Local Client Model

Empirical Study: DBMSs

PostgreSQL

Client-Server Model

Simultaneous Read/Write

SQLite

Local Client Model

Empirical Study: DBMSs

PostgreSQL

Client-Server Model

Simultaneous Read/Write

SQLite

Local Client Model

Locking on Write

Empirical Study: DBMSs

PostgreSQL

Client-Server Model

Simultaneous Read/Write

SQLite

Local Client Model

Locking on Write

Prevents Parallel Approaches

Results

Results

- Median of repetitions

Results

- Median of repetitions
- Lower-is-better metric

Results

- Median of repetitions
- Lower-is-better metric
- Split by...

Results

- Median of repetitions
- Lower-is-better metric
- Split by...

  ...case study

Results

- Median of repetitions
- Lower-is-better metric
- Split by...

  ...case study

  ...DBMS

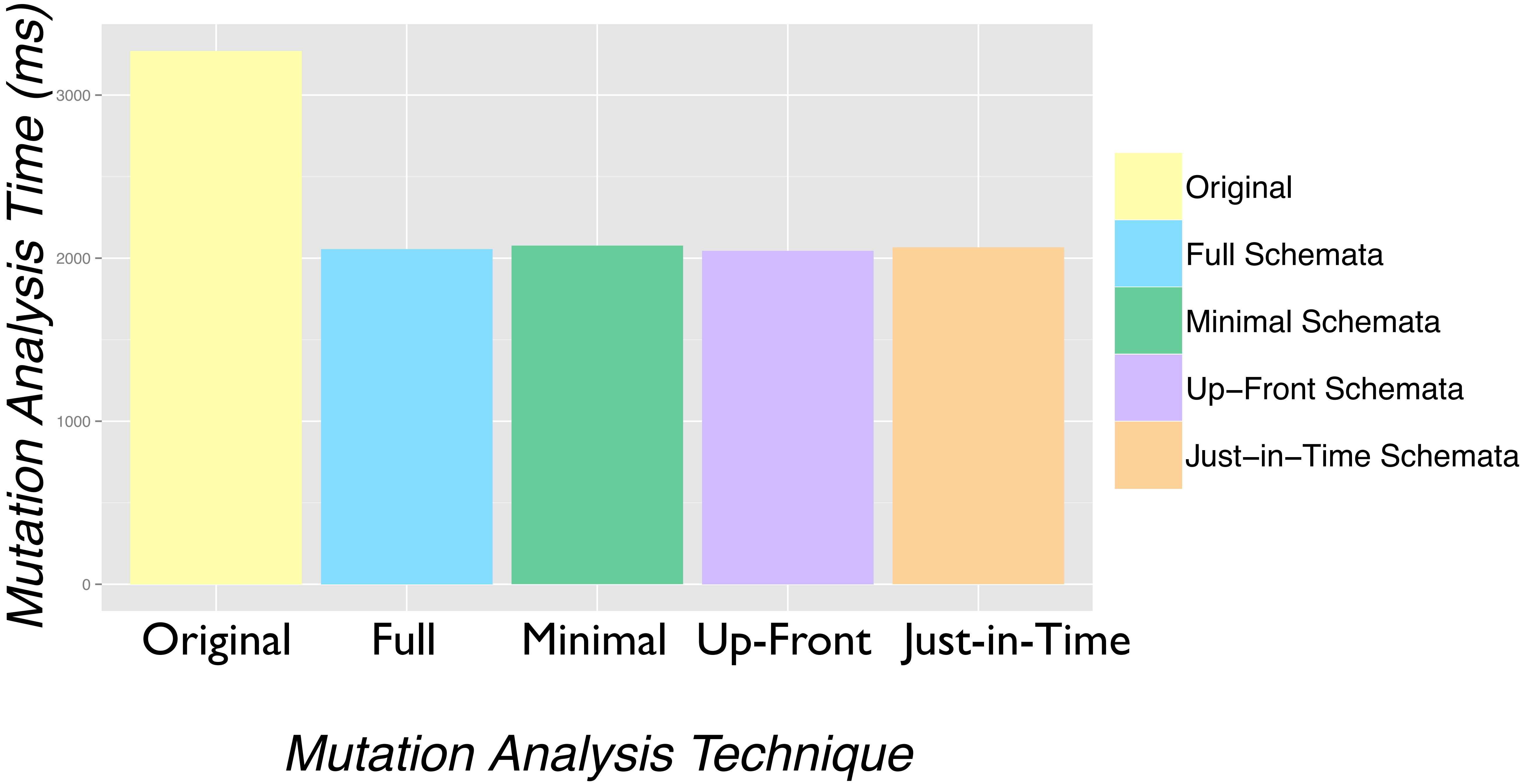
Results

- Median of repetitions
- Lower-is-better metric
- Split by...

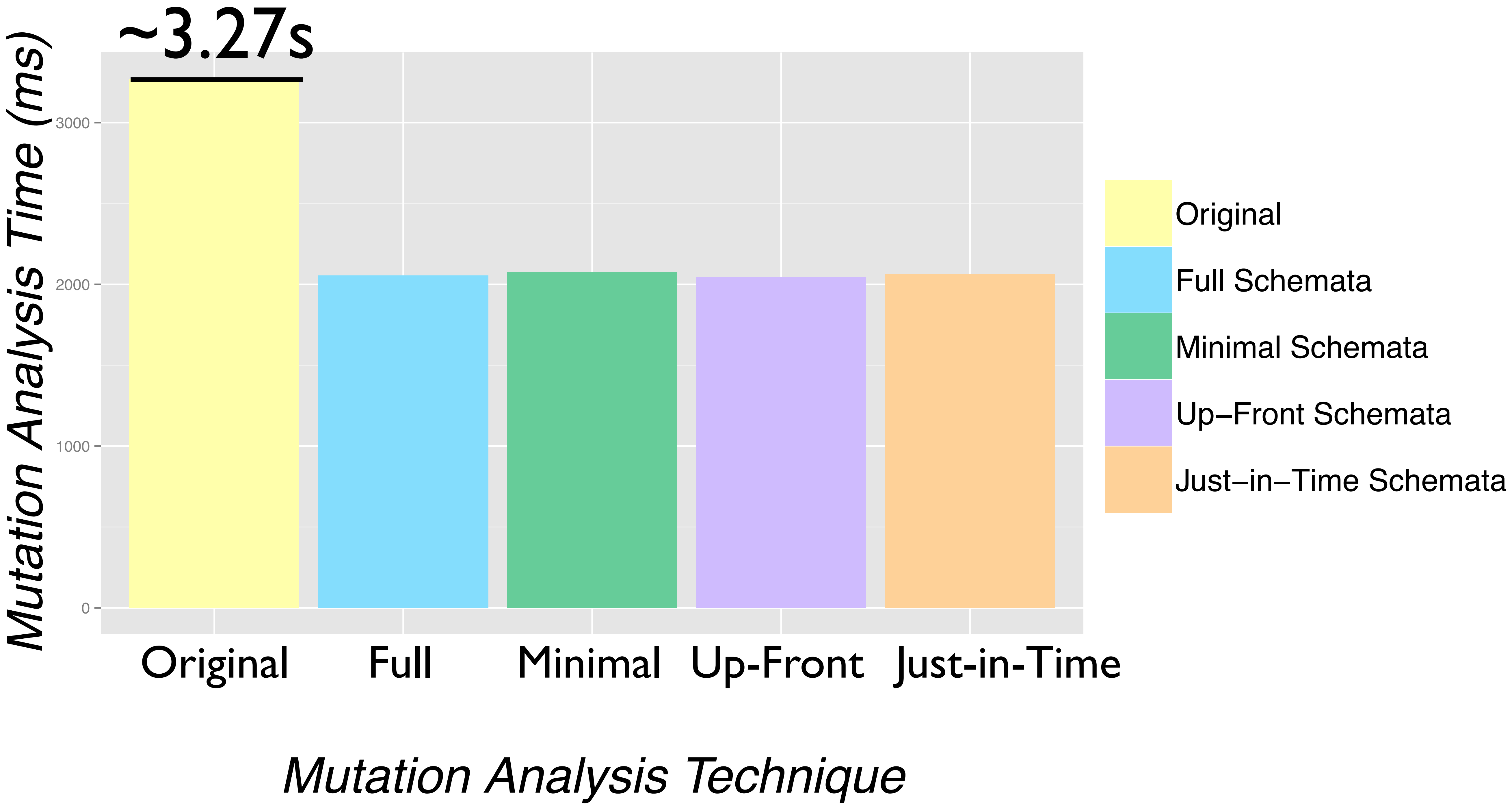
  ...case study

  ...DBMS
- Full details in paper (including statistics)

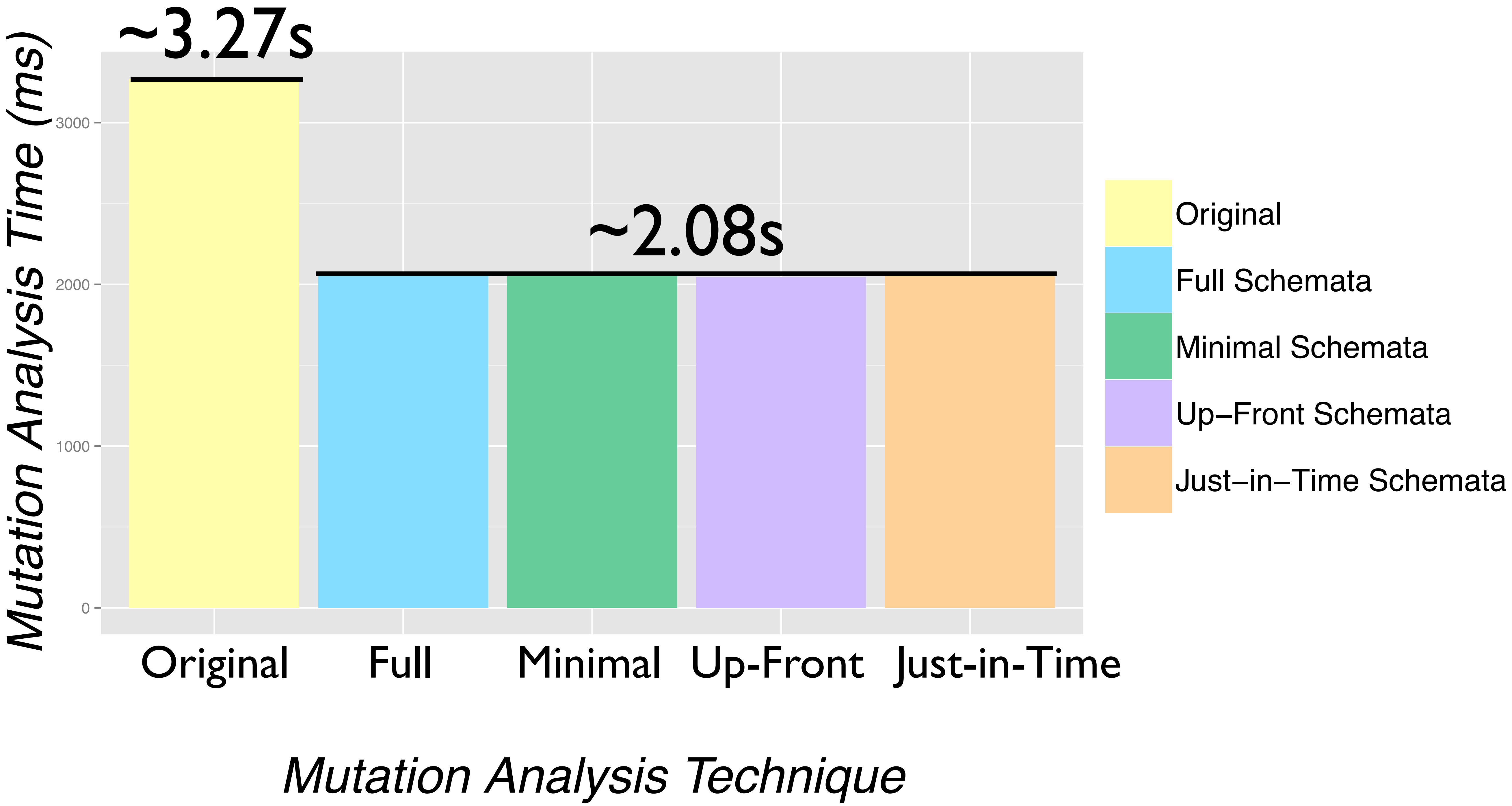
Postgres – Cloc



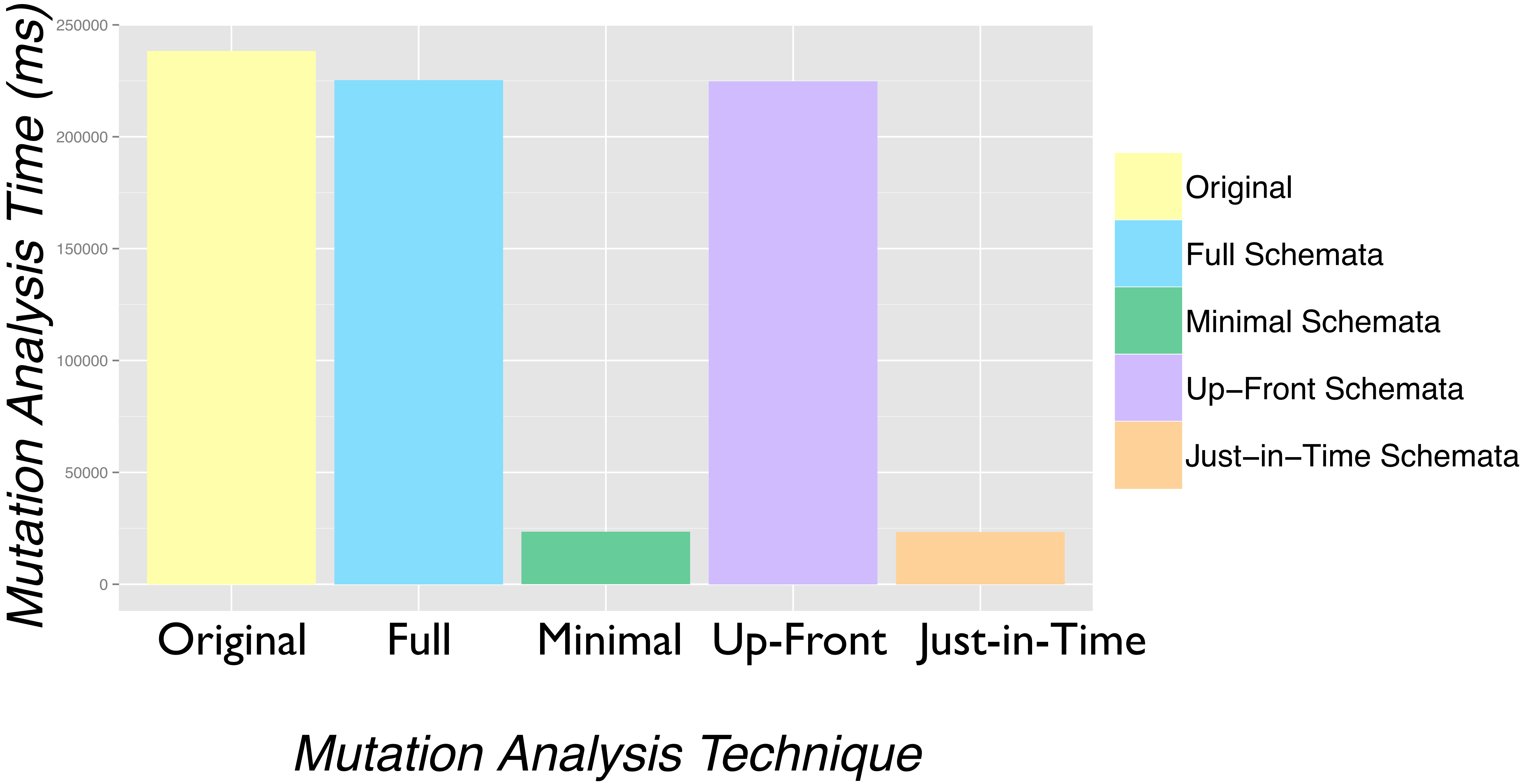
Postgres – Cloc



Postgres – Cloc

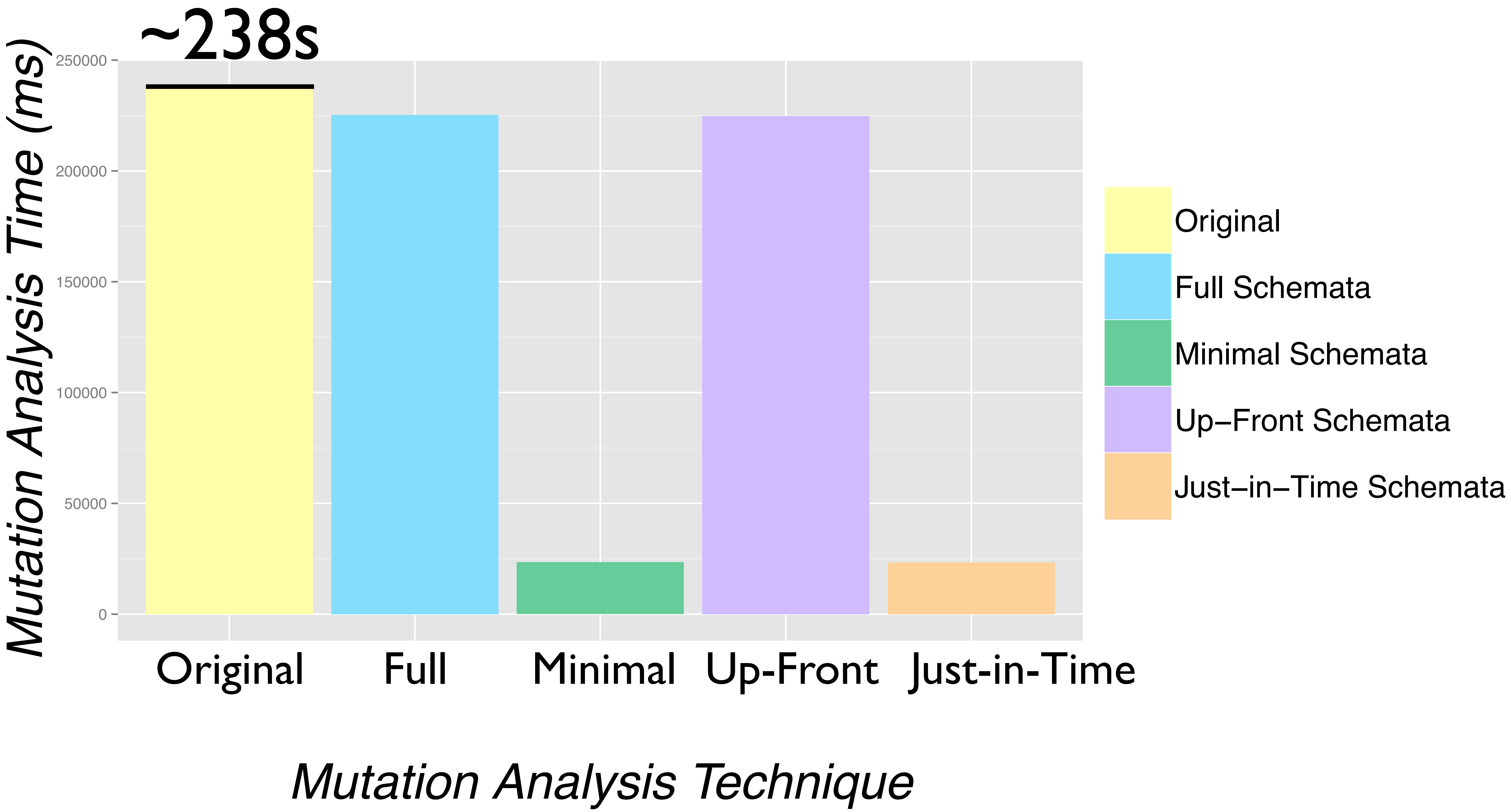


Postgres – RiskIt



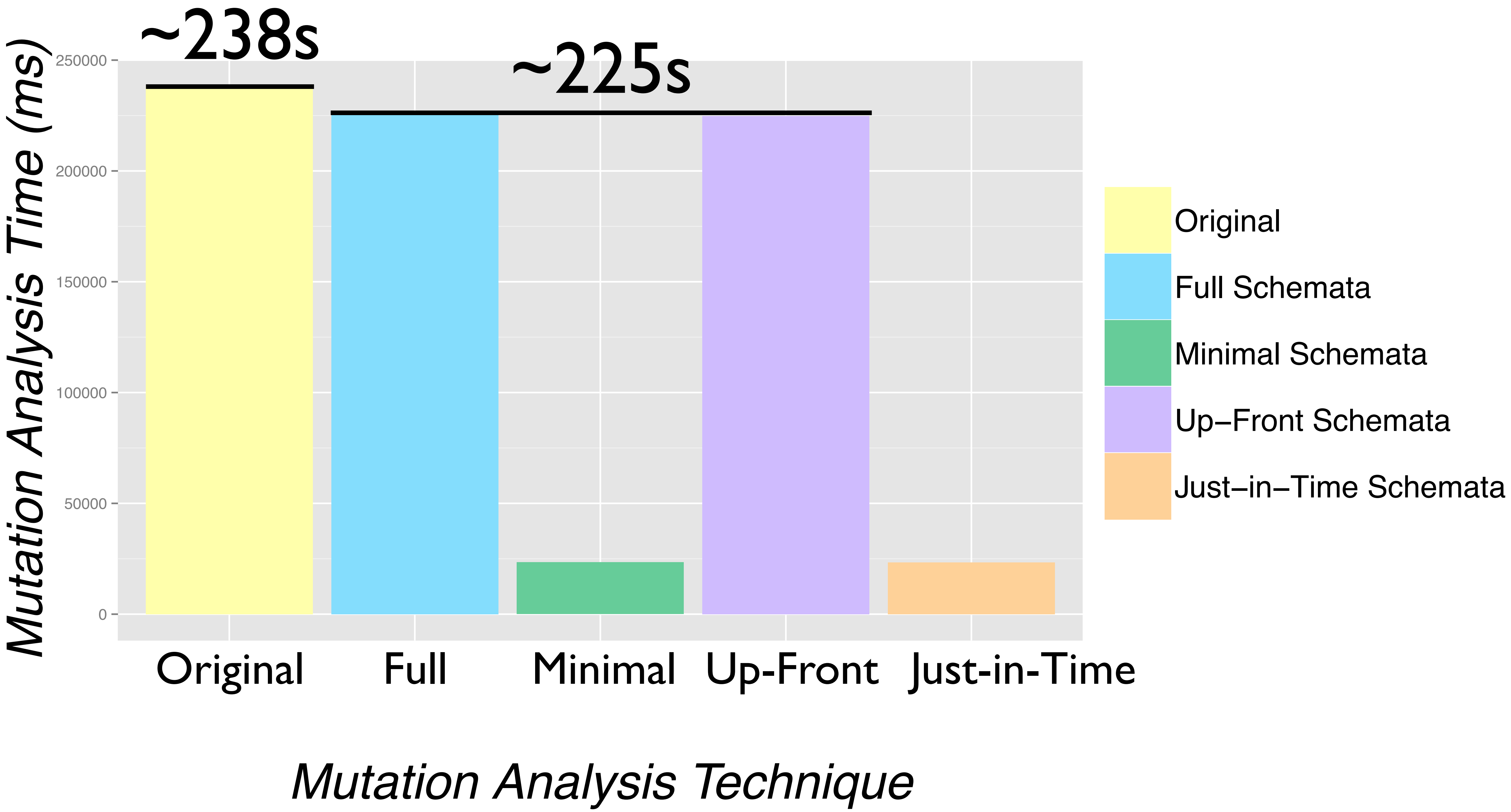
Chris J. Wright - c.wright@dcs.shef.ac.uk

Postgres – RiskIt



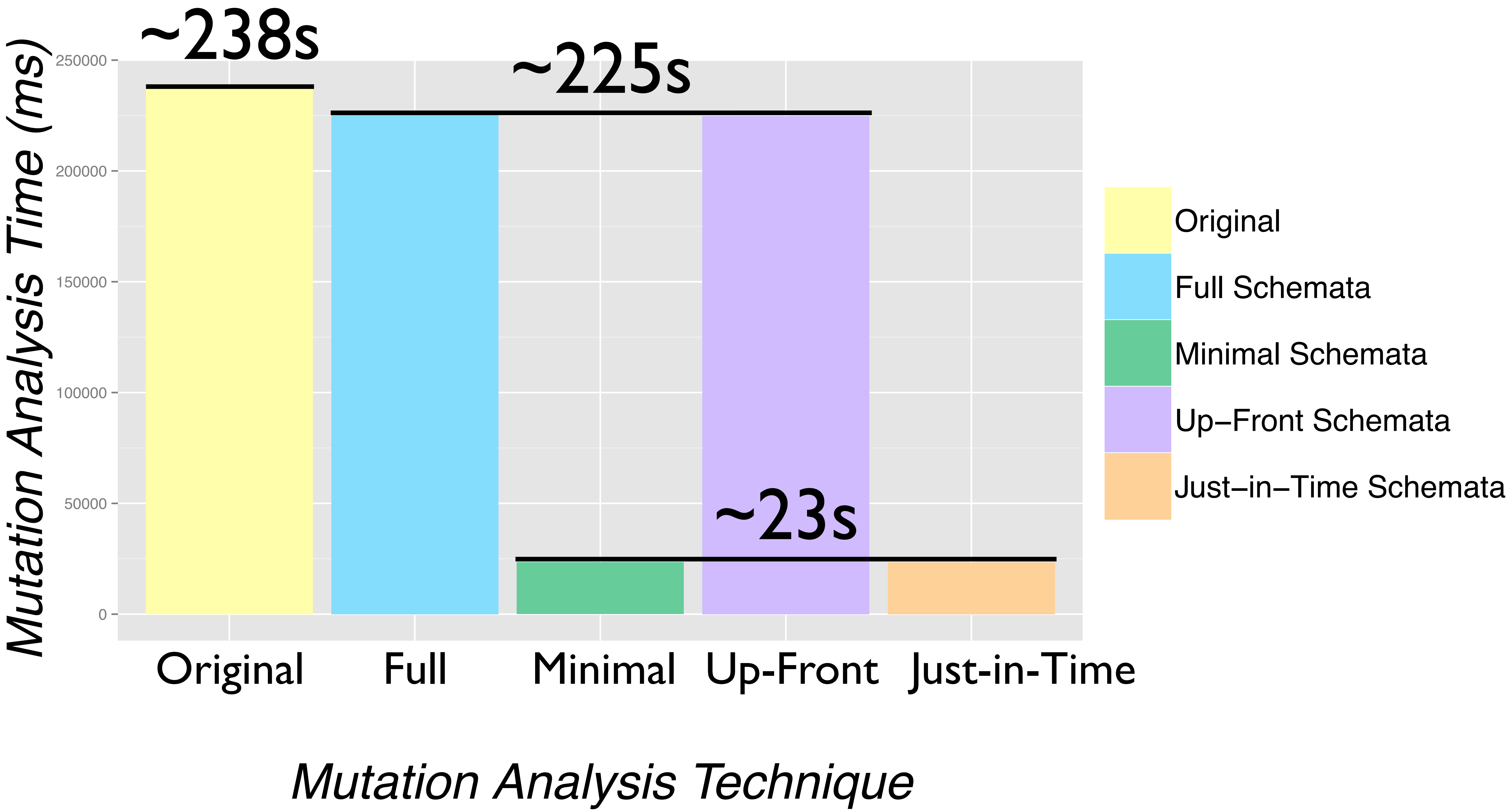
Chris J. Wright - c.wright@dcs.shef.ac.uk

Postgres – RiskIt



Chris J. Wright - c.wright@dcs.shef.ac.uk

Postgres – RiskIt



Chris J. Wright - c.wright@dcs.shef.ac.uk

Results – Postgres

Results – Postgres

‘Full Schemata’

Results – Postgres

‘Full Schemata’

Improvement decreases with larger schemas

Results – Postgres

‘Full Schemata’

Improvement decreases with larger schemas

‘Minimal Schemata’

Results – Postgres

‘Full Schemata’

Improvement decreases with larger schemas

‘Minimal Schemata’

Consistently faster, scales very well

Results – Postgres

‘Full Schemata’

Improvement decreases with larger schemas

‘Minimal Schemata’

Consistently faster, scales very well

‘Just-in-Time Schemata’

Results – Postgres

‘Full Schemata’

Improvement decreases with larger schemas

‘Minimal Schemata’

Consistently faster, scales very well

‘Just-in-Time Schemata’

Consistently faster, scales very well

Results – Postgres

‘Full Schemata’

Improvement decreases with larger schemas

‘Minimal Schemata’

Consistently faster, scales very well

‘Just-in-Time Schemata’

Consistently faster, scales very well

‘Up-Front Schemata’

Results – Postgres

‘Full Schemata’

Improvement decreases with larger schemas

‘Minimal Schemata’

Consistently faster, scales very well

‘Just-in-Time Schemata’

Consistently faster, scales very well

‘Up-Front Schemata’

Improvement decreases with larger schemas

Results – Postgres

‘Full Schemata’

Improvement decreases with larger schemas

‘Minimal Schemata’

Consistently faster, scales very well

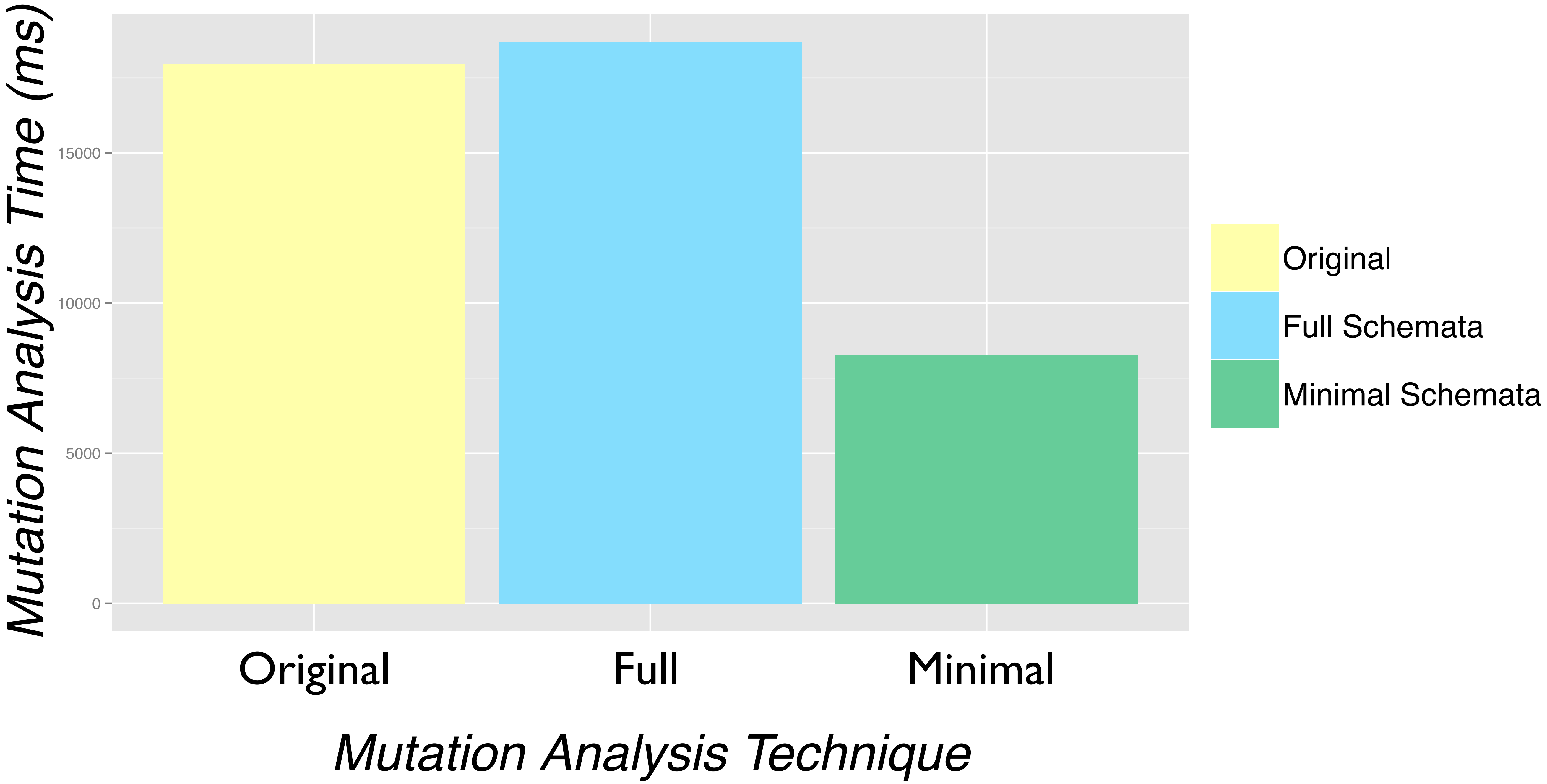
‘Just-in-Time Schemata’

Consistently faster, scales very well

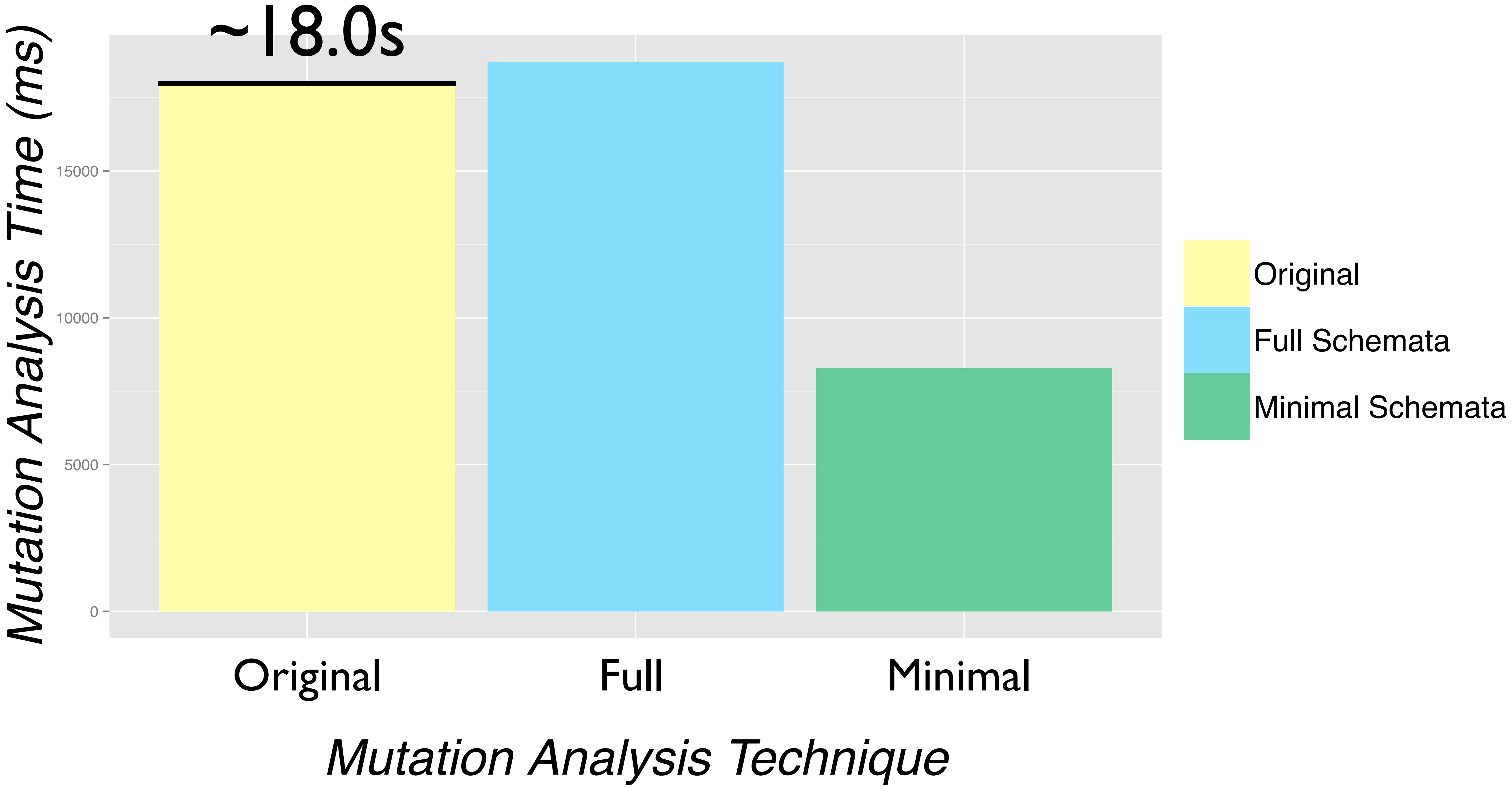
‘Up-Front Schemata’

Improvement decreases with larger schemas

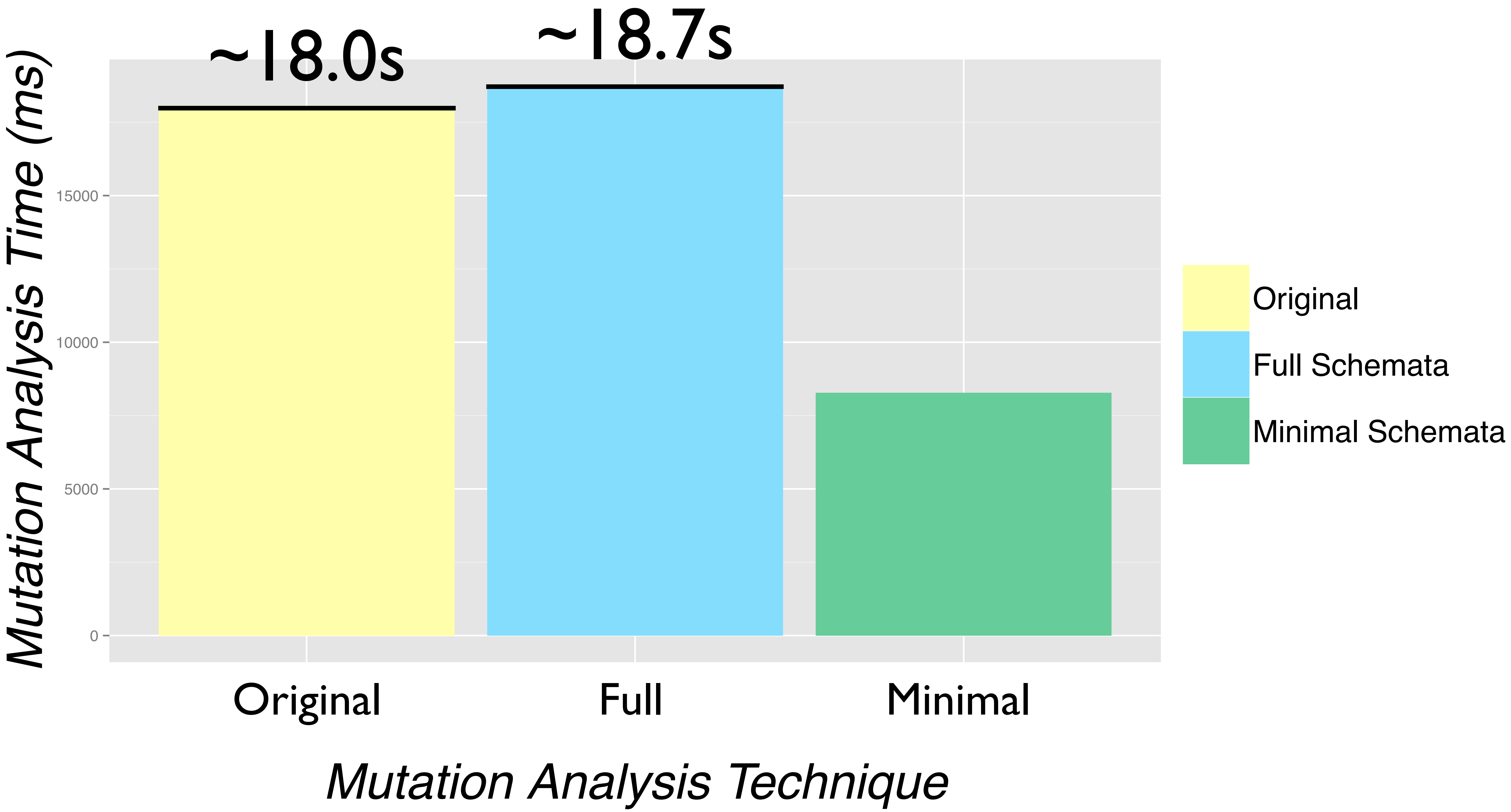
SQLite – Cloc


Mutation Analysis Time (ms)
15000
10000
5000
0
Original
Full
Minimal
Mutation Analysis Technique
Original
Full Schemata
Minimal Schemata

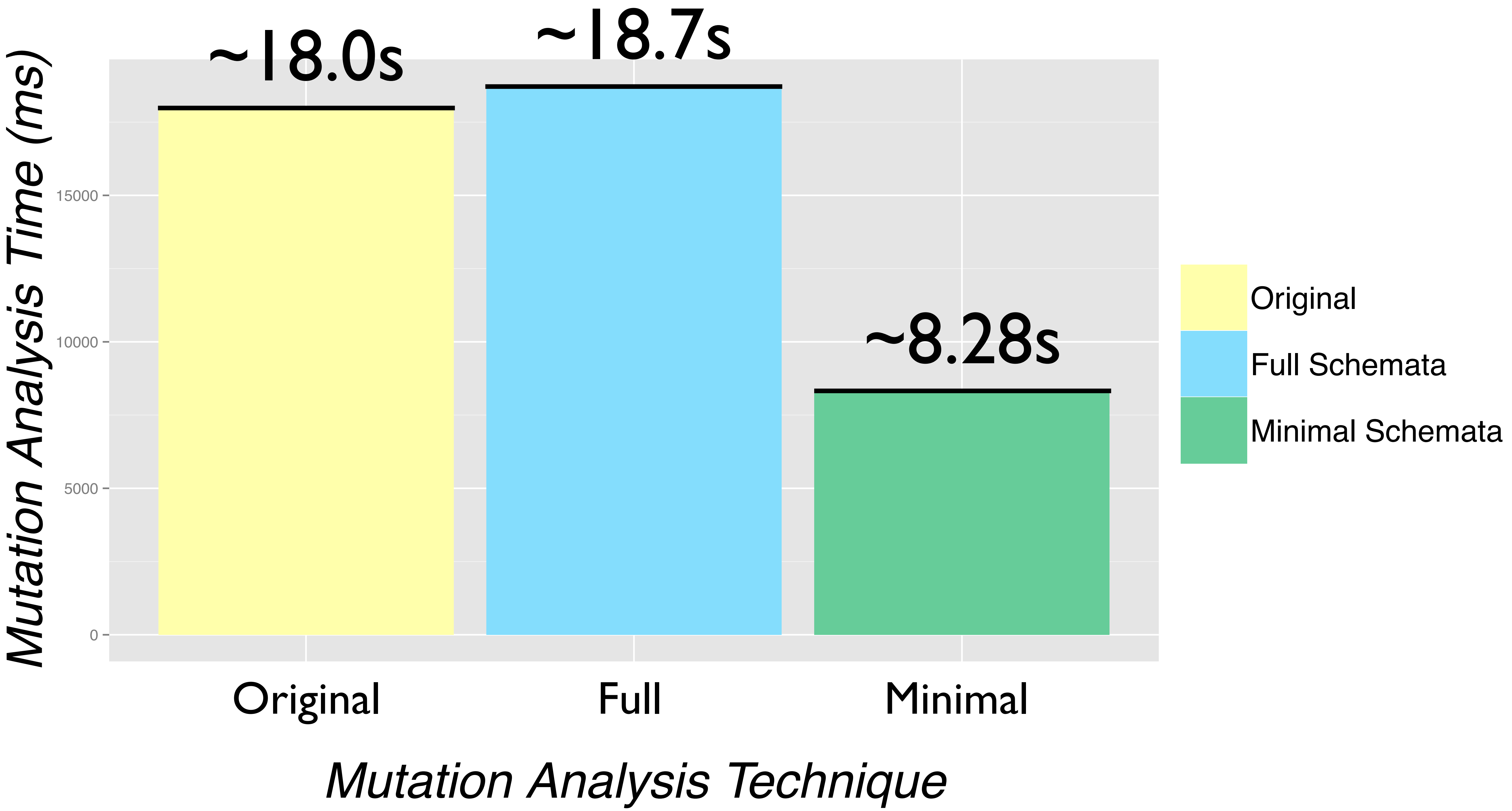
SQLite – Cloc


~18.0s
Mutation Analysis Time (ms)
15000
10000
5000
0
Original
Full
Minimal
Mutation Analysis Technique
Original
Full Schemata
Minimal Schemata

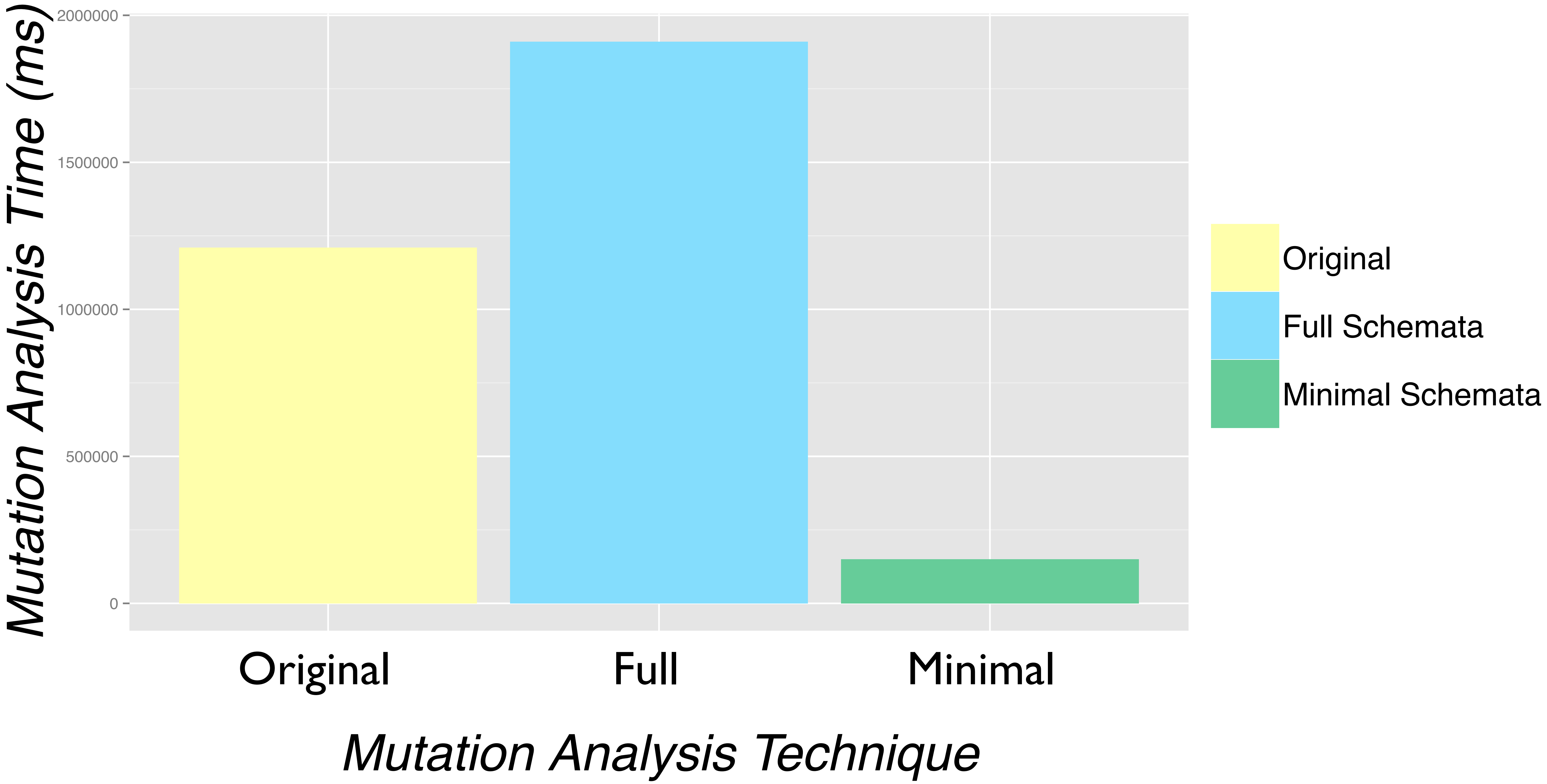
SQLite – Cloc


~18.0s
~18.7s
Mutation Analysis Time (ms)
15000
10000
5000
0
Original
Full
Minimal
Mutation Analysis Technique
Original
Full Schemata
Minimal Schemata

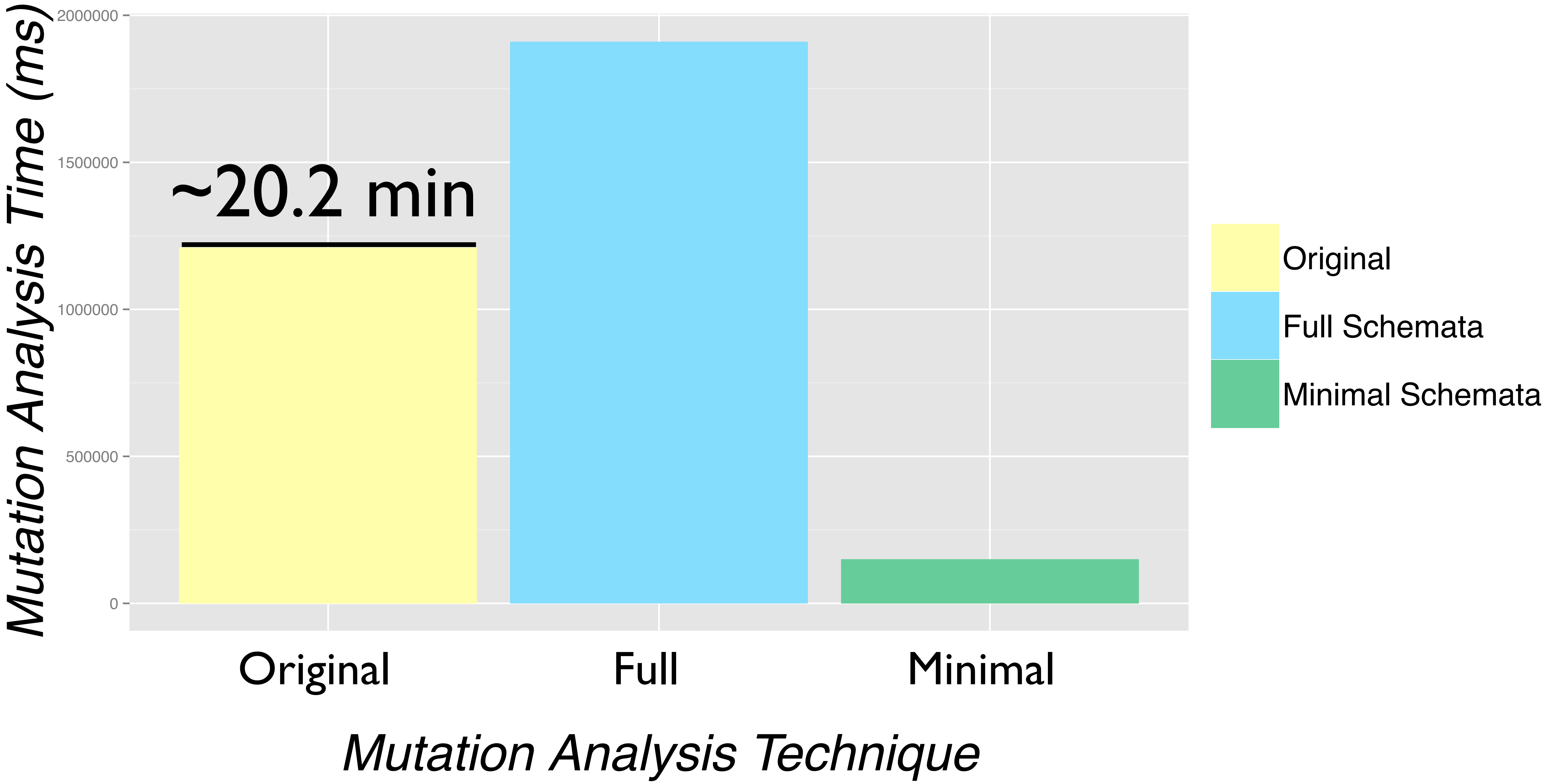
SQLite – Cloc


~18.0s
~18.7s
~8.28s
Mutation Analysis Time (ms)
15000
10000
5000
0
Original
Full
Minimal
Mutation Analysis Technique
Original
Full Schemata
Minimal Schemata

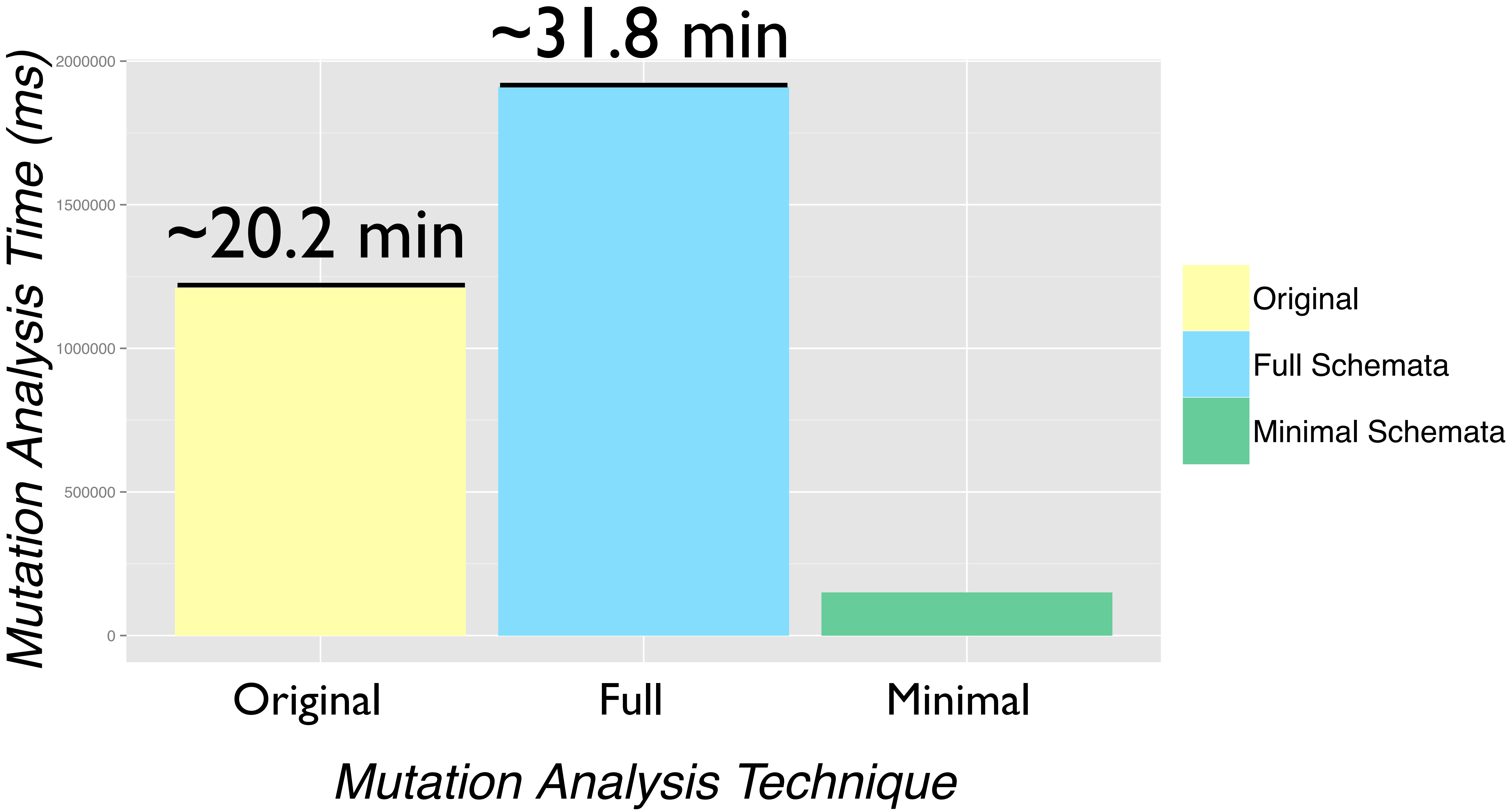
SQLite – RiskIt



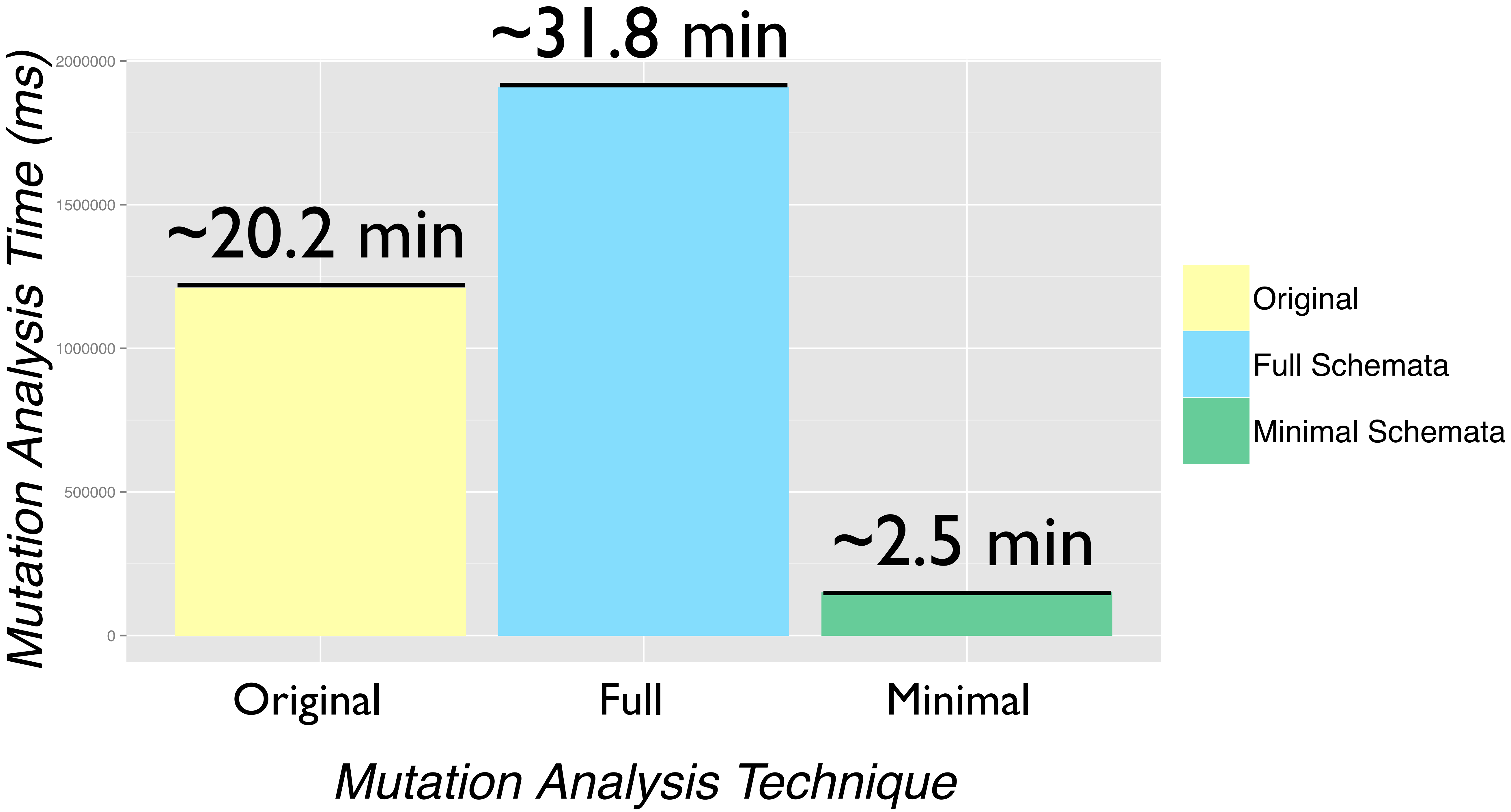
SQLite – RiskIt



SQLite – RiskIt



SQLite – RiskIt



Chris J. Wright - c.wright@dcs.shef.ac.uk

Results – SQLite

Results – SQLite

‘Full Schemata’

Results – SQLite

‘Full Schemata’

Increasingly worsened with larger schemas

Results – SQLite

‘Full Schemata’

Increasingly worsened with larger schemas

‘Minimal Schemata’

Results – SQLite

‘Full Schemata’

Increasingly worsened with larger schemas

‘Minimal Schemata’

Consistently faster, scales very well

Results – SQLite

‘Full Schemata’

Increasingly worsened with larger schemas

‘Minimal Schemata’

Consistently faster, scales very well

Future Work & Limitations

Future Work & Limitations

Case Studies

Future Work & Limitations

Case Studies

DBMSs

Future Work & Limitations

Case Studies

DBMSs

Detailed Timing

Future Work & Limitations

Case Studies

DBMSs

Detailed Timing

Approaches

Future Work & Limitations

Case Studies

DBMSs

Detailed Timing

Approaches

Parallel Configurations

Future Work & Limitations

- Case Studies
- DBMSs
- Detailed Timing
- Approaches
- Parallel Configurations
- Test Suite

Test Suite Generation

Test Suite Generation

- SchemaAnalyst tool

Test Suite Generation

- SchemaAnalyst tool
- Gregory Kapfhammer

Test Suite Generation

- SchemaAnalyst tool
- Gregory Kapfhammer
- Tuesday 11:00am, Research & Industrial Track

Conclusions

Conclusions

- Mutant Schemata and Parallelisation can both reduce the cost of mutation analysis

Conclusions

- Mutant Schemata and Parallelisation can both reduce the cost of mutation analysis
- The 'Minimal Schemata' approach...

Conclusions

- Mutant Schemata and Parallelisation can both reduce the cost of mutation analysis
- The ‘Minimal Schemata’ approach...

...consistently faster than original

Conclusions

- Mutant Schemata and Parallelisation can both reduce the cost of mutation analysis
- The 'Minimal Schemata' approach...

...consistently faster than original

...gives a reduction of up to 10x

Conclusions

- Mutant Schemata and Parallelisation can both reduce the cost of mutation analysis
- The ‘Minimal Schemata’ approach...

...consistently faster than original

...gives a reduction of up to 10x

...scales very well (for our case studies)

Conclusions

- Mutant Schemata and Parallelisation can both reduce the cost of mutation analysis
- The 'Minimal Schemata' approach...

...consistently faster than original

...gives a reduction of up to 10x

...scales very well (for our case studies)

...doesn't require parallel DBMS access

Conclusions

- Mutant Schemata and Parallelisation can both reduce the cost of mutation analysis
- The 'Minimal Schemata' approach...

  ...consistently faster than original

  ...gives a reduction of up to 10x

  ...scales very well (for our case studies)

  ...doesn't require parallel DBMS access
- Website: http://schemaanalyst.org/

Conclusions

- Mutant Schemata and Parallelisation can both reduce the cost of mutation analysis
- The 'Minimal Schemata' approach...

  ...consistently faster than original

  ...gives a reduction of up to 10x

  ...scales very well (for our case studies)

  ...doesn't require parallel DBMS access
- Website: http://schemaanalyst.org/