Better data quality with constraints

INTRODUCTION TO RELATIONAL DATABASES IN SQL

SQL

Timo Grossenbacher
Data Journalist

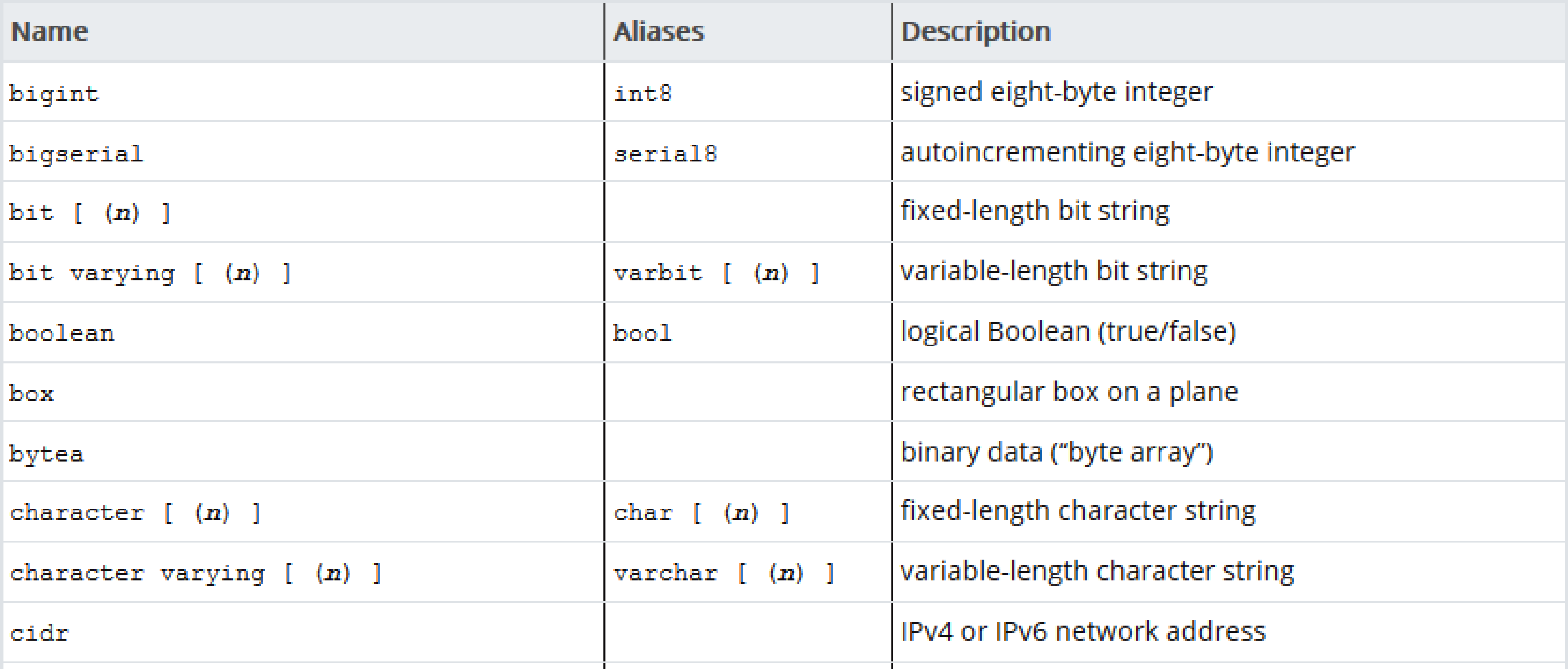
Integrity constraints

1. **Attribute constraints**, e.g. data types on columns (Chapter 2)
2. **Key constraints**, e.g. primary keys (Chapter 3)
3. **Referential integrity constraints**, enforced through foreign keys (Chapter 4)

Why constraints?

- Constraints give the data structure
- Constraints help with consistency, and thus data quality
- Data quality is a business advantage / data science prerequisite
- Enforcing is difficult, but PostgreSQL helps

Data types as attribute constraints



Name	Aliases	Description
`bigint`	`int8`	signed eight-byte integer
`bigserial`	`serial8`	autoincrementing eight-byte integer
`bit [ (n) ]`		fixed-length bit string
`bit varying [ (n) ]`	`varbit [ (n) ]`	variable-length bit string
`boolean`	`bool`	logical Boolean (true/false)
`box`		rectangular box on a plane
`bytea`		binary data ("byte array")
`character [ (n) ]`	`char [ (n) ]`	fixed-length character string
`character varying [ (n) ]`	`varchar [ (n) ]`	variable-length character string
`cidr`		IPv4 or IPv6 network address

From the **PostgreSQL documentation**.

Dealing with data types (casting)

```
CREATE TABLE weather (
 temperature integer,
 wind_speed text);
SELECT temperature * wind_speed AS wind_chill
FROM weather;
```

```
operator does not exist: integer * text
HINT: No operator matches the given name and argument type(s).
You might need to add explicit type casts.
```

```
SELECT temperature * CAST(wind_speed AS integer) AS wind_chill
FROM weather;
```

Let's practice!

INTRODUCTION TO RELATIONAL DATABASES IN SQL

Working with data types

INTRODUCTION TO RELATIONAL DATABASES IN SQL

SQL

Timo Grossenbacher
Data Journalist

Working with data types

- Enforced on columns (i.e. attributes)
- Define the so-called "domain" of a column
- Define what operations are possible
- Enfore consistent storage of values

The most common types

- `text` : character strings of any length
- `varchar [ (x) ]` : a maximum of `n` characters
- `char [ (x) ]` : a fixed-length string of `n` characters
- `boolean` : can only take three states, e.g. `TRUE` , `FALSE` and `NULL` (unknown)

From the **PostgreSQL documentation**.

The most common types (cont'd.)

- `date` , `time` and `timestamp` : various formats for date and time calculations
- `numeric` : arbitrary precision numbers, e.g. `3.1457`
- `integer` : whole numbers in the range of `-2147483648` and `+2147483647`

From the **PostgreSQL documentation**.

Specifying types upon table creation

```
CREATE TABLE students (
 ssn integer,
 name varchar(64),
 dob date,
 average_grade numeric(3, 2), -- e.g. 5.54
 tuition_paid boolean
);
```

Alter types after table creation

```sql
ALTER TABLE students
ALTER COLUMN name
TYPE varchar(128);
```

```sql
ALTER TABLE students
ALTER COLUMN average_grade
TYPE integer
-- Turns 5.54 into 6, not 5, before type conversion
USING ROUND(average_grade);
```

Let's apply this!

INTRODUCTION TO RELATIONAL DATABASES IN SQL

The not-null and unique constraints

INTRODUCTION TO RELATIONAL DATABASES IN SQL

SQL

Timo Grossenbacher
Data Journalist

The not-null constraint

- Disallow `NULL` values in a certain column
- Must hold true for the current state
- Must hold true for any future state

What does NULL mean?

- unknown
- does not exist
- does not apply
- ...

What does NULL mean? An example

```
CREATE TABLE students (
 ssn integer not null,
 lastname varchar(64) not null,
 home_phone integer,
 office_phone integer
);
```

```
NULL != NULL
```

How to add or remove a not-null constraint

When creating a table...

```
CREATE TABLE students (
 ssn integer not null,
 lastname varchar(64) not null,
 home_phone integer,
 office_phone integer
);
```

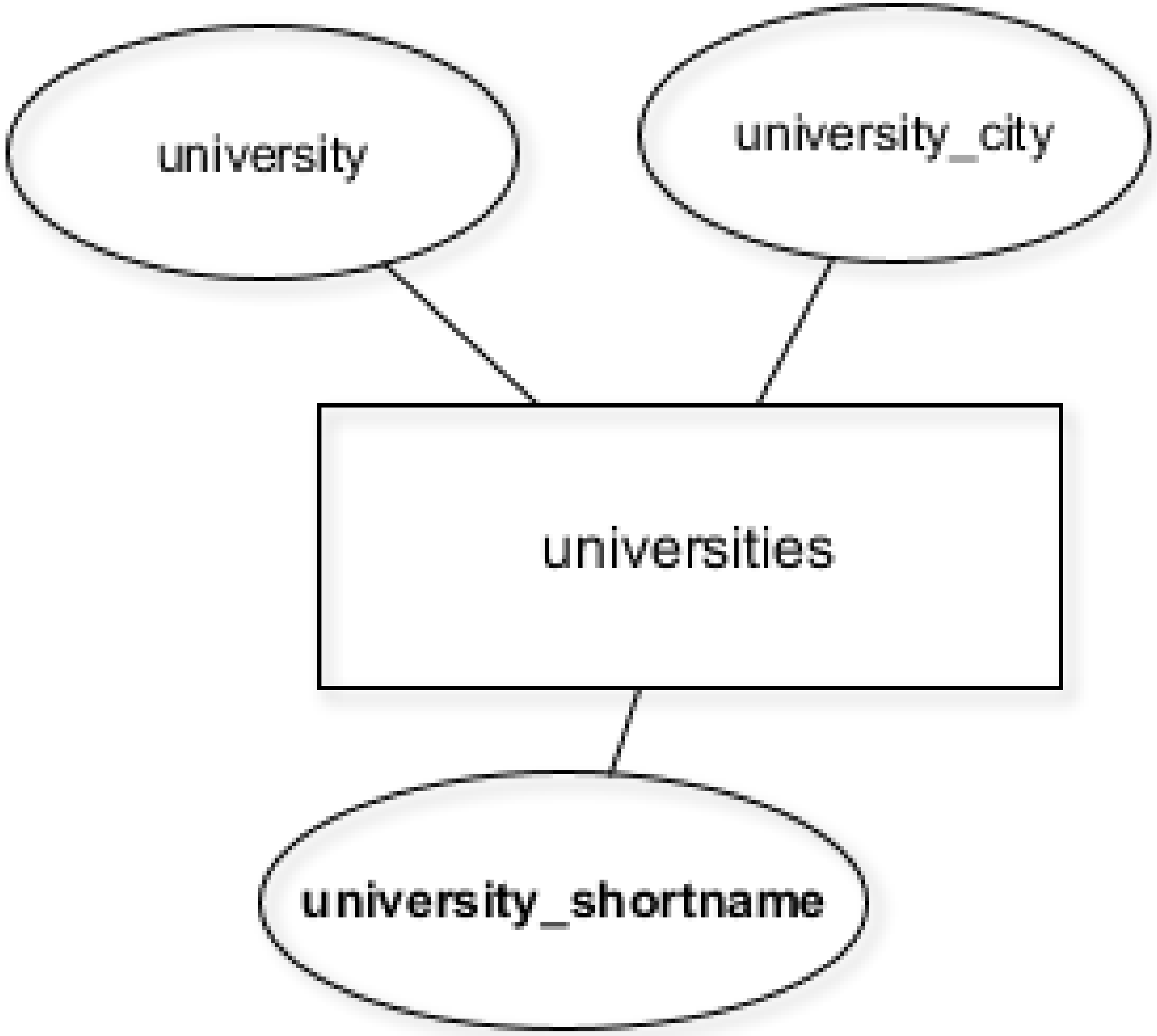
After the table has been created...

```
ALTER TABLE students
ALTER COLUMN home_phone
SET NOT NULL;
```

```
ALTER TABLE students
ALTER COLUMN ssn
DROP NOT NULL;
```

The unique constraint

- Disallow duplicate values in a column
- Must hold true for the current state
- Must hold true for any future state



Adding unique constraints

```
CREATE TABLE table_name (
 column_name UNIQUE
);
```

```
ALTER TABLE table_name
ADD CONSTRAINT some_name UNIQUE(column_name);
```

Let's apply this to the database!

INTRODUCTION TO RELATIONAL DATABASES IN SQL