SQL DDL DML

DDL – Data Definition Language – Create, Remove and Modify Database Structures

- `Create`
- `Alter`
- `Drop`

DML – Data Manipulation Language – Create, Remove and Modify Data Values

- `Insert`
- `Update`
- `Delete`
- `Truncate`

SQL Create

The CREATE statement

- Create a new table or view
  - Defines the table NAME and its COLUMNS
  - Defines the DATA TYPE, LENGTH for each column
  - Defines the CONSTRAINTS for each column
- Usually we DROP the table (if it exists) before you create it

SQL Create

CREATE statement

```
CREATE TABLE <table name>
   (<column name> <DATATYPE(L)>,
    <column name> <DATATYPE(L)> NOT NULL,
    <column name> <DATATYPE>    NOT NULL Default 0,
    <column name> <DATATYPE> CONSTRAINT <constraint name> <TYPE>,
    <column name> <DATATYPE(L)>)
```

SQL Create

CREATE statement

```
CREATE TABLE <table name>
(
    column DATATYPE(L),
    column DATATYPE(L) NOT NULL,
    column DATATYPE(L) NOT NULL Default 0,
    column DATATYPE(L) CONSTRAINT <constraint name> TYPE,
    column DATATYPE(L)
)
```

SQL Create

DROP Statement

```
DROP TABLE IF EXISTS "alanparadise/nw"."shippers"
```

SQL Create

CREATE statement

```
CREATE TABLE "alanparadise/nw"."shippers"
(
     ShipperID   int           NOT NULL ,
     CompanyName varchar(40) NOT NULL ,
     Phone       varchar(20) NOT NULL DEFAULT '0'
);
```

SQL Create – NOT NULL Constraint

The "NOT NULL" constraint

The database software will NOT allow a row to be inserted if the column has no value

However, if the column is missing a value in the INSERT, the DEFAULT option can provide a default value, allowing the insert

(More later on CONSTRAINTS ...)

SQL Create – Inserts

```
INSERT INTO  "alanparadise/nw"."shippers"  VALUES (1, 'Speedy Express', '(503) 555-9831');
INSERT INTO  "alanparadise/nw"."shippers"  VALUES (2, 'United Package', '(503) 555-3199');
INSERT INTO  "alanparadise/nw"."shippers"  VALUES (3, 'Federal Shipping');
```

SQL Create - Data Types

The DATA TYPES available in the CREATE statement depend on your database engine.

bit.io uses an underlying PostgreSQL database engine

PostgreSQL offers all of the following data types:

https://www.postgresql.org/docs/12/datatype.html

SQL Create - Data Types

Some Commonly Used PostgreSQL Data Types:

```
int, bigint, smallint

boolean

char(n), varchar(n)

text

date

timestamp

decimal(x,y), float, real

JSON
```

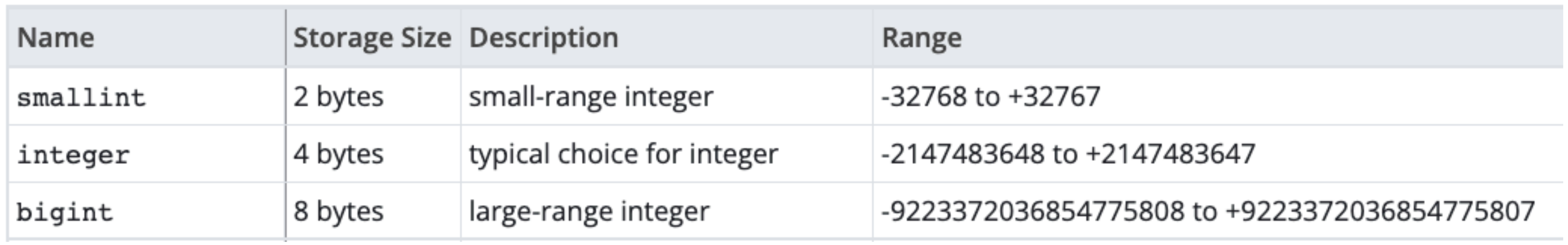
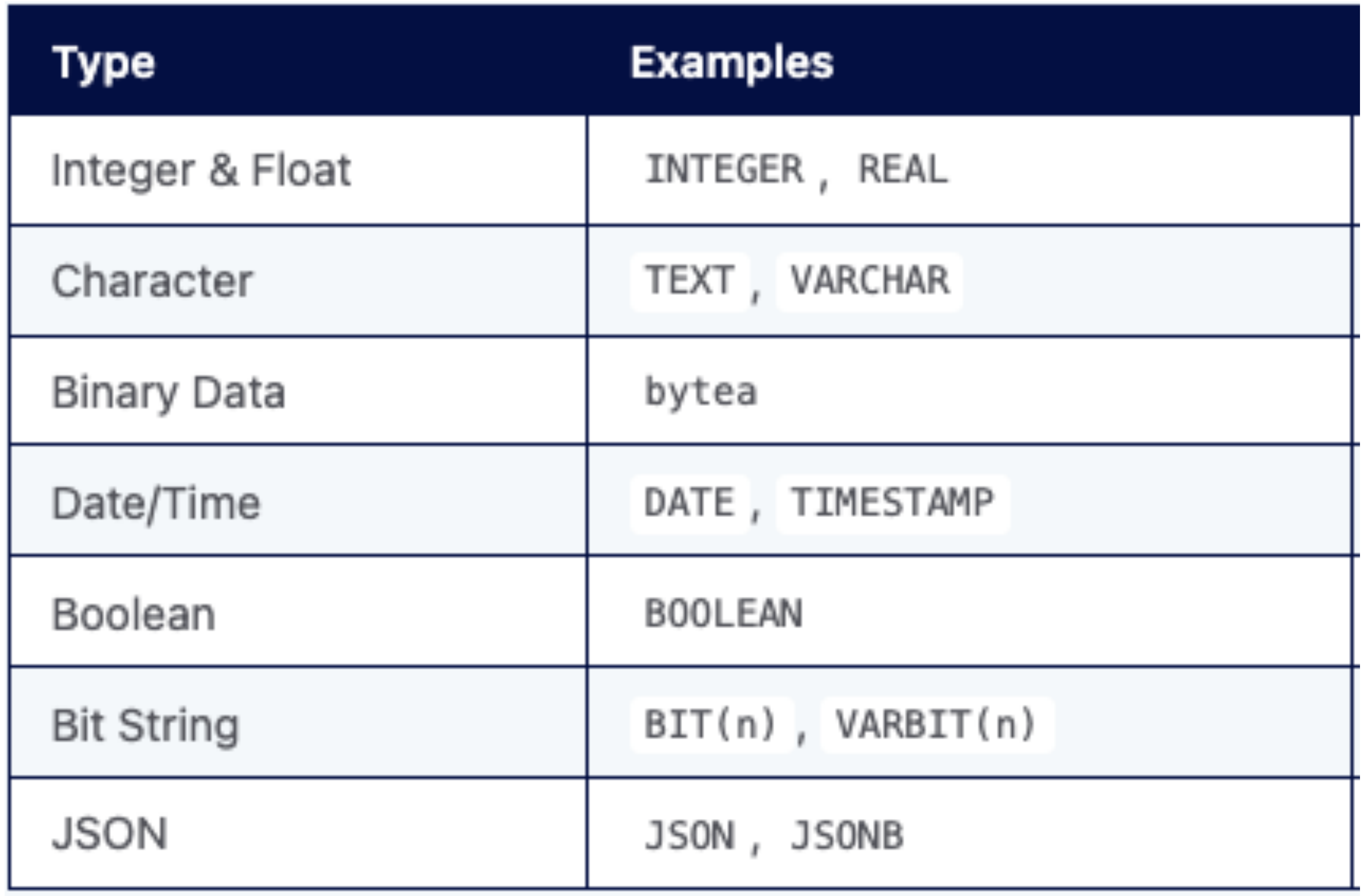


Table 8.2. Numeric Types

Name	Storage Size	Description	Range
`smallint`	2 bytes	small-range integer	-32768 to +32767
`integer`	4 bytes	typical choice for integer	-2147483648 to +2147483647
`bigint`	8 bytes	large-range integer	-9223372036854775808 to +9223372036854775807

SQL Create - Data Types

However, at this time, bit.io supports only:



Type	Examples
Integer & Float	INTEGER , REAL
Character	TEXT , VARCHAR
Binary Data	bytea
Date/Time	DATE , TIMESTAMP
Boolean	BOOLEAN
Bit String	BIT(n) , VARBIT(n)
JSON	JSON , JSONB