SQL Fundamentals

Jenna Olsen

1. Day 1

➔ **(Relational) Databases**
The what & the why

➔ **Creating & Populating Tables**
The how

➔ **Technologies & Platforms**
The where

Database

➔ Collection

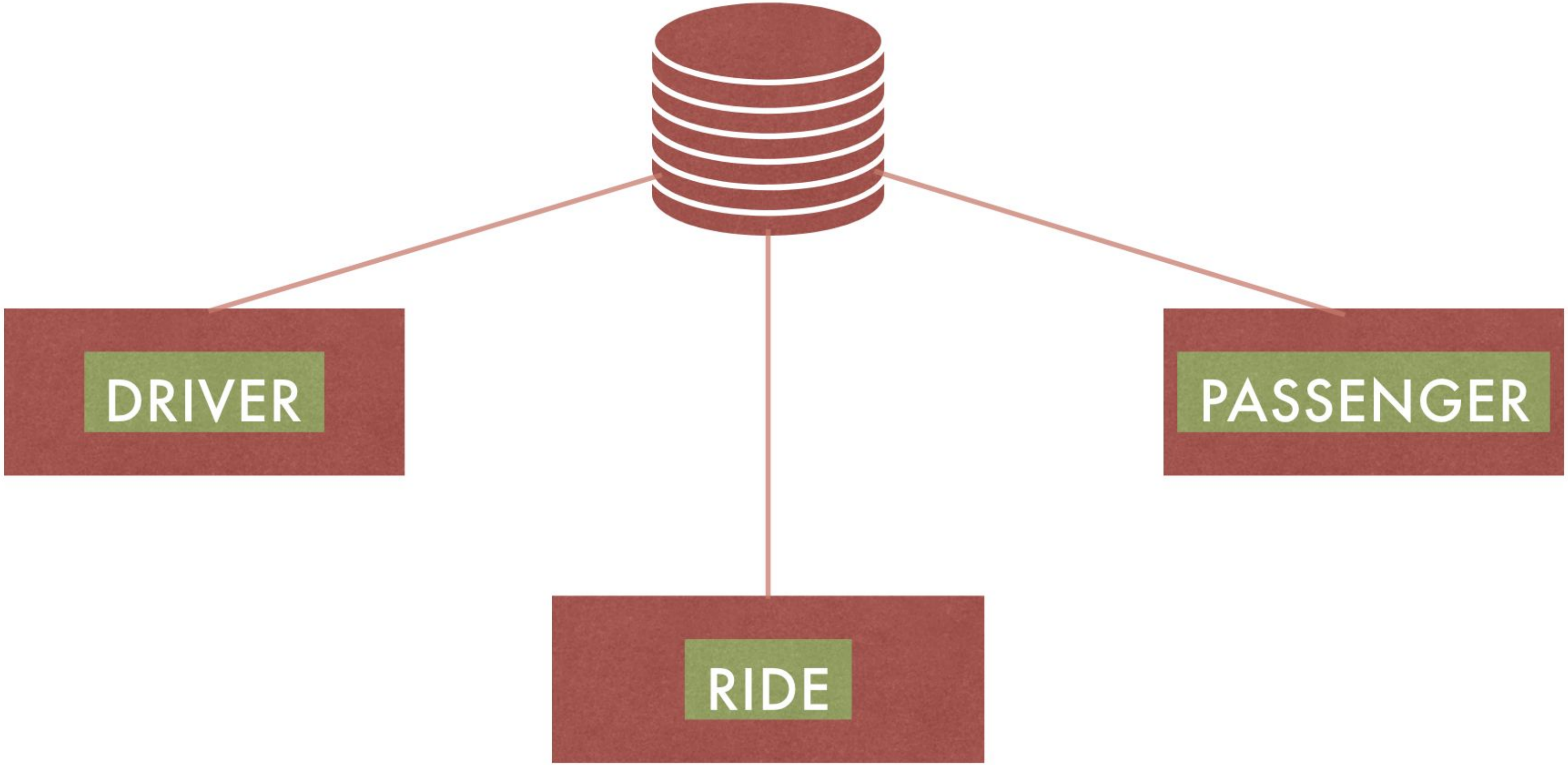
➔ Organized

➔ DBMS

Relational Database

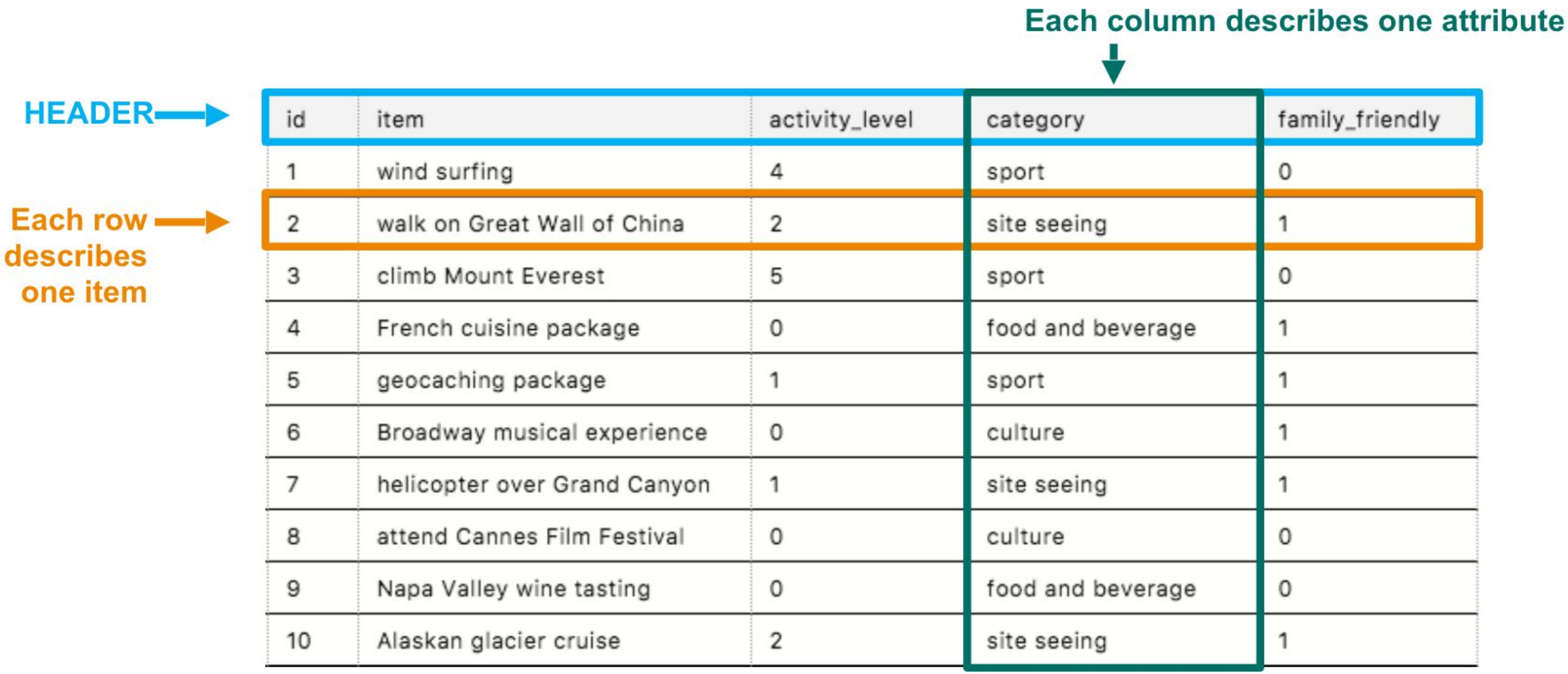
- Tables/Views/Schemas
- Key - Relationship
  - Primary
    - Requirements
  - Foreign

Ride-Share Data Example



Relational Database Example

Relational table detailing activities offered by Company XYZ below



Each column describes one attribute

HEADER

Each row describes one item

id	item	activity_level	category	family_friendly
1	wind surfing	4	sport	0
2	walk on Great Wall of China	2	site seeing	1
3	climb Mount Everest	5	sport	0
4	French cuisine package	0	food and beverage	1
5	geocaching package	1	sport	1
6	Broadway musical experience	0	culture	1
7	helicopter over Grand Canyon	1	site seeing	1
8	attend Cannes Film Festival	0	culture	0
9	Napa Valley wine tasting	0	food and beverage	0
10	Alaskan glacier cruise	2	site seeing	1

How many languages do you need to know to communicate with the (relational) data of the world?

1. SQL “sequel”

Structured Query Language

a. Domain-specific
b. Designed to
    i. Query relational data
    ii. Manage data
    iii. Interact with RDBMS
c. Especially useful with structured & related data

Databases are *not* (select one):

a. Made of “tables”
b. A programming language
c. A collection of data

Primary Keys (select multiple):

a. Must contain a unique value for each row
b. Must be a number
c. Cannot be NULL
d. Must be sorted
e. Must be in the left-most column

Creating a Table

Using a retail inventory data example, we can create three columns containing integers or text:

```sql
CREATE TABLE inventory (
    id INTEGER PRIMARY KEY,
    name TEXT,
    quantity INTEGER
);
```

Creating Table with More Data Types

Add new columns in inventory table for tracking: vendor unit price, date of last shipment, and whether or not we should reorder:

```
CREATE TABLE inventory (
    id INTEGER PRIMARY KEY,
    name VARCHAR(100),
    quantity INTEGER,
    vendor_unit_price MONEY,
    last_shipment DATE,
    reorder BIT
);
```

Switched from TEXT to VARCHAR(100); No more than 100 characters permitted

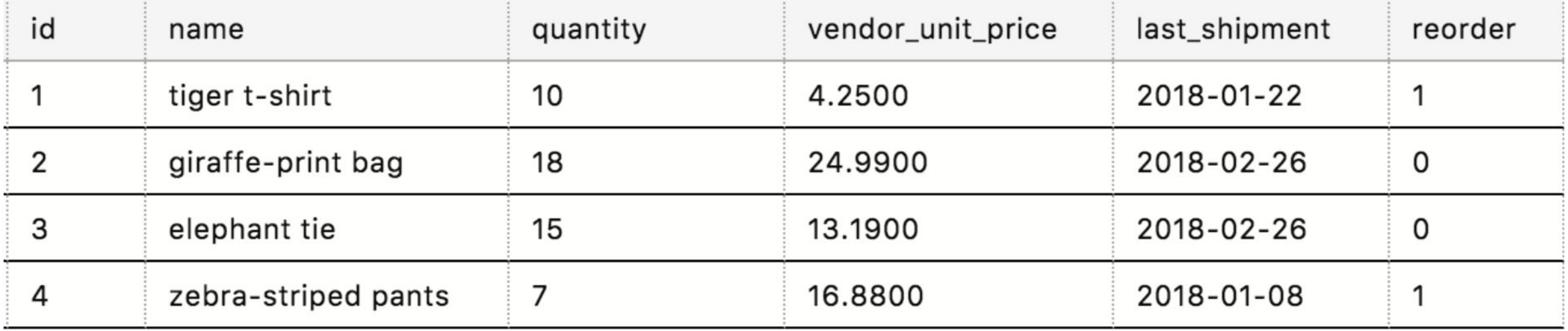
Creating Table with More Data Types

Now INSERT INTO commands contain information about unit price, last shipment, and reordering:

```
INSERT INTO inventory VALUES (1, ‘tiger t-shirt’, 10, 4.25,
    ‘2018-01-22’, ‘TRUE’);
INSERT INTO inventory VALUES (2, ‘giraffe-print bag’, 18, 24.99,
    ‘2018-02-26’, ‘FALSE’);
INSERT INTO inventory VALUES (3, ‘elephant tie’, 15, 13.19,
    ‘2018-02-26’, ‘FALSE’);
INSERT INTO inventory VALUES (4, ‘zebra-striped pants’, 7,
     16.88, ‘2018-01-08’, ‘TRUE’);
```

Creating Table with More Data Types

Resulting table created with new data types:

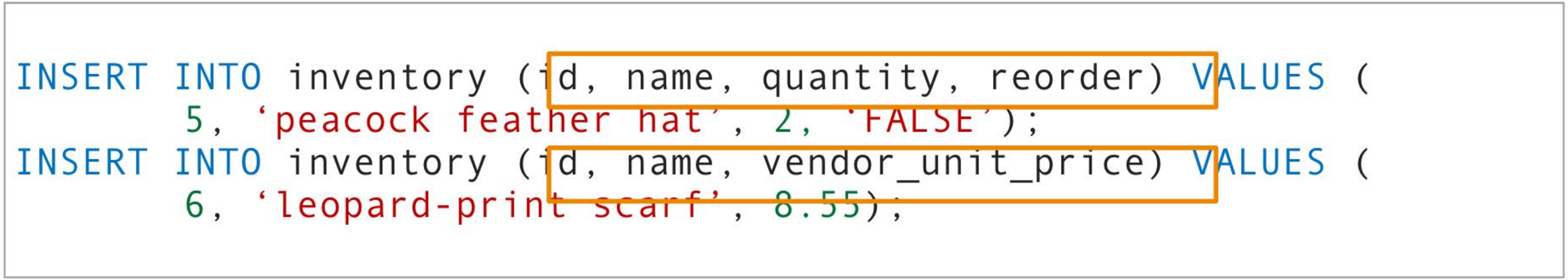


id	name	quantity	vendor_unit_price	last_shipment	reorder
1	tiger t-shirt	10	4.2500	2018-01-22	1
2	giraffe-print bag	18	24.9900	2018-02-26	0
3	elephant tie	15	13.1900	2018-02-26	0
4	zebra-striped pants	7	16.8800	2018-01-08	1

Note: ‘TRUE’ and ‘FALSE’ are converted to 1 and 0, respectively, since they are BIT data types

Creating Table with Missing Values

Add two more rows with some information missing:

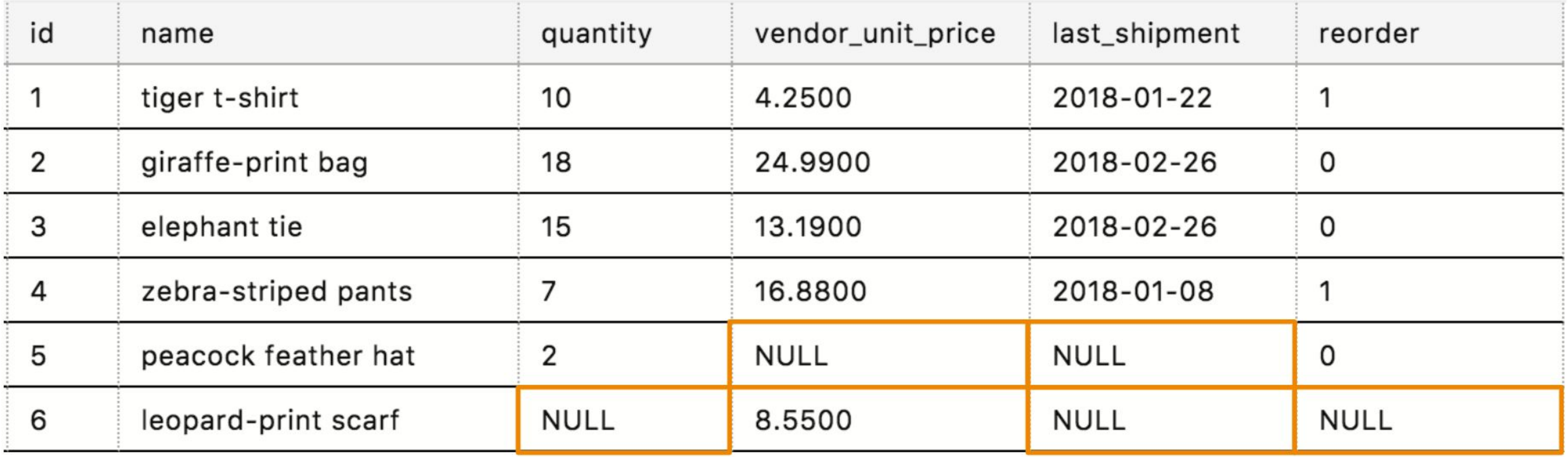


```
INSERT INTO inventory (id, name, quantity, reorder) VALUES (
        5, ‘peacock feather hat’, 2, ‘FALSE’);
INSERT INTO inventory (id, name, vendor_unit_price) VALUES (
        6, ‘leopard-print scarf’, 8.55);
```

We must include column names after table name so SQL knows which attributes have values!

Creating Another Table

SQL inserts NULL for any missing value:



id	name	quantity	vendor_unit_price	last_shipment	reorder
1	tiger t-shirt	10	4.2500	2018-01-22	1
2	giraffe-print bag	18	24.9900	2018-02-26	0
3	elephant tie	15	13.1900	2018-02-26	0
4	zebra-striped pants	7	16.8800	2018-01-08	1
5	peacock feather hat	2	NULL	NULL	0
6	leopard-print scarf	NULL	8.5500	NULL	NULL

Updating a Table

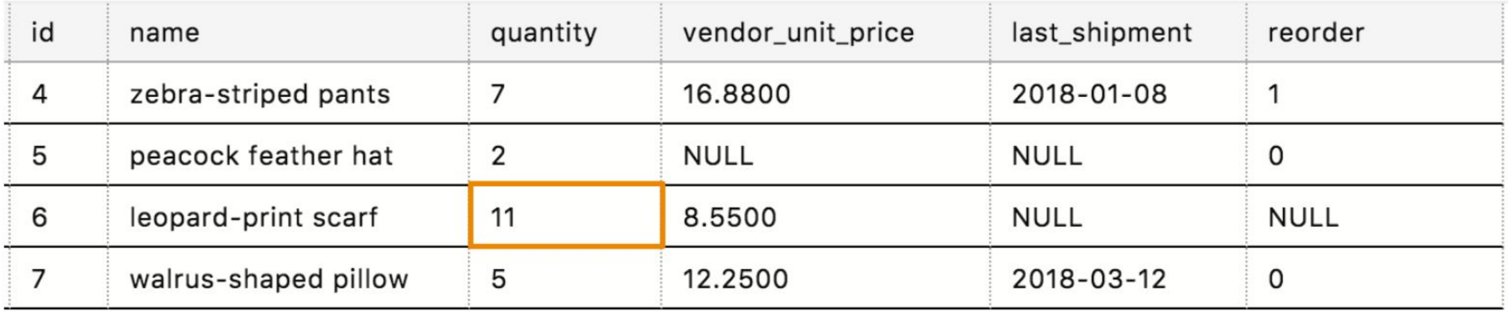
Company has 11 leopard-print scarves in stock. Update inventory table.

```
UPDATE inventory SET quantity=11 WHERE id=6;
```

Name of table to be updated

Column to change and value to insert

Specify row to change!



id	name	quantity	vendor_unit_price	last_shipment	reorder
4	zebra-striped pants	7	16.8800	2018-01-08	1
5	peacock feather hat	2	NULL	NULL	0
6	leopard-print scarf	11	8.5500	NULL	NULL
7	walrus-shaped pillow	5	12.2500	2018-03-12	0

Deleting Rows

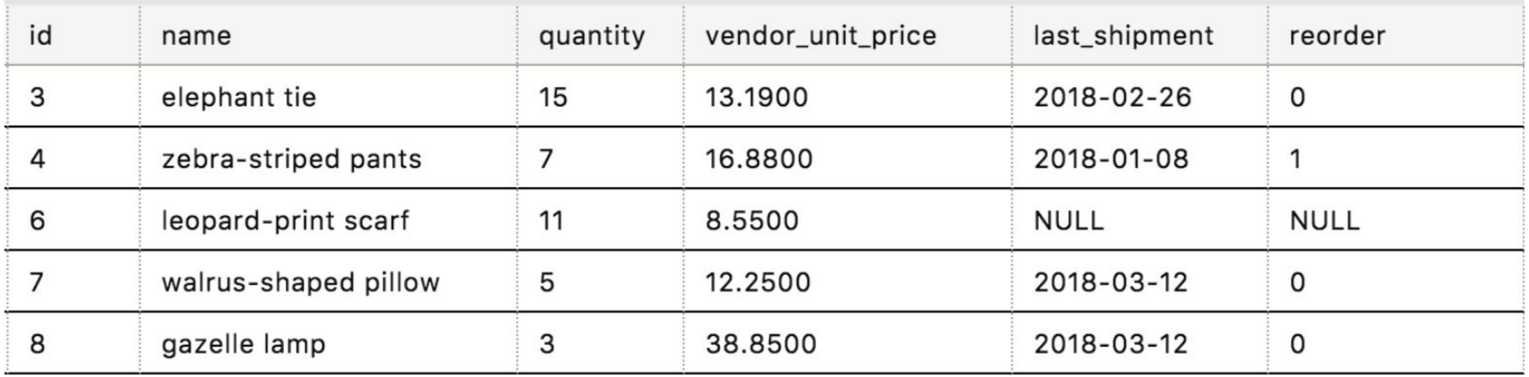
Company no longer offers peacock hat. Delete row from inventory table.

```
DELETE FROM inventory WHERE id=5;
```

Name of table that contains row

Specify row to change!

Item #5 no longer in table



id	name	quantity	vendor_unit_price	last_shipment	reorder
3	elephant tie	15	13.1900	2018-02-26	0
4	zebra-striped pants	7	16.8800	2018-01-08	1
6	leopard-print scarf	11	8.5500	NULL	NULL
7	walrus-shaped pillow	5	12.2500	2018-03-12	0
8	gazelle lamp	3	38.8500	2018-03-12	0

Deleting Tables and Columns

To complete erase the inventory table:

```
DROP TABLE inventory;
```

To drop (or add) a column, use ALTER TABLE command:

```
ALTER TABLE inventory DROP COLUMN reorder;
```

Note: Dropping a column changes the table’s schema.

Rename Table

To rename the inventory table:

```
ALTER TABLE inventory
  RENAME TO supplies;
```

Adding a New Column

Add the name of my current vendor for each item in inventory table.

```
ALTER TABLE inventory ADD vendor_name VARCHAR(100);
```

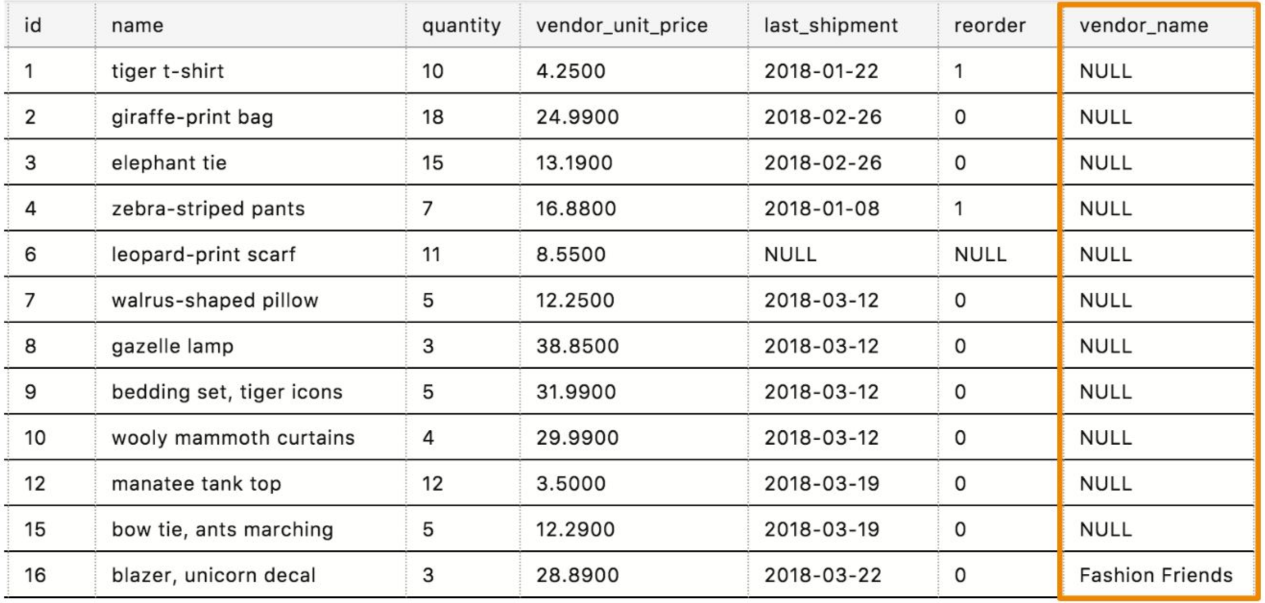
Now can add vendor’s name when logging new items into inventory table:

```
INSERT INTO inventory VALUES (
      16, ‘blazer, unicorn decal’, 3, 28.89,
      ‘2018-03-22’, 0, ‘Fashion Friends’
);
```

Include vendor’s name

Adding a New Column

Inventory Table



id	name	quantity	vendor_unit_price	last_shipment	reorder	vendor_name
1	tiger t-shirt	10	4.2500	2018-01-22	1	NULL
2	giraffe-print bag	18	24.9900	2018-02-26	0	NULL
3	elephant tie	15	13.1900	2018-02-26	0	NULL
4	zebra-striped pants	7	16.8800	2018-01-08	1	NULL
6	leopard-print scarf	11	8.5500	NULL	NULL	NULL
7	walrus-shaped pillow	5	12.2500	2018-03-12	0	NULL
8	gazelle lamp	3	38.8500	2018-03-12	0	NULL
9	bedding set, tiger icons	5	31.9900	2018-03-12	0	NULL
10	wooly mammoth curtains	4	29.9900	2018-03-12	0	NULL
12	manatee tank top	12	3.5000	2018-03-19	0	NULL
15	bow tie, ants marching	5	12.2900	2018-03-19	0	NULL
16	blazer, unicorn decal	3	28.8900	2018-03-22	0	Fashion Friends

New column added

NULL values for previous items

Can set default when creating column to avoid NULLs

```
ALTER TABLE inventory ADD vendor_name VARCHAR(100)
    NOT NULL DEFAULT(‘Trend Zoo, Inc.’);
```

DB Browser for SQLite

http://sqlitebrowser.org/

Soccer Data Set

https://www.kaggle.com/hugomathien/soccer/downloads/soccer.zip

Time for the Challenges

Challenges file in Github repo

https://github.com/thisismetis/lol18_sql3/blob/master/Challenges_Week1.md