EST. 1785 UNB

UNIVERSITY OF NEW BRUNSWICK

Faculty of Computer Science

CS1103

Introduction to Databases

Lecture 2

Relational Database Fundamentals

Part II – Table Creation

Learning Objectives

By the end of this topic, you will be able to:

PART I:

- Describe how a relational database is organized
- Design a simple database schema

PART II:

- Use Data Definition Language (DDL) to create a database
- Insert data into database tables
- Display database table contents

Let's start by creating these tables from Part I

album

album_id	album_name	artist_name
100	Greatest Hits	Queen
101	Greatest Hits	Boston
102	Rumours	Fleetwood Mac
103	Second Helping	Lynyrd Skynyrd

track

album_id	track_number	song_title	length
100	1	Killer Queen	3:01
100	2	Bohemian Rhapsody	5:55
101	3	More Than a Feeling	null
101	12	Rock and Roll Band	null
102	1	Second Hand News	2:55
102	2	Dreams	4:32
103	1	Sweet Home Alabama	4:43
103	8	Call Me The Breeze	5:09

Data Definition Language (DDL)

album

album_id	album_name	artist_name
100	Greatest Hits	Queen
101	Greatest Hits	Boston
102	Rumours	Fleetwood Mac
103	Second Helping	Lynyrd Skynyrd

```
create table album
(   <define all columns & keys>
);

CREATE TABLE album
(   <define all columns & keys>
);
```

Names of tables, columns, etc. are case-sensitive

- album   Album   ALBUM   -- are different names

Common naming conventions:

- Lowercase speeds typing
- No spaces – use underscores instead
- Letters only – avoid digits
- Max 64 characters in a name
- Self explanatory meaning
- Be consistent – either singular or plural

A Complete 'create table' Statement

album

album_id	album_name	artist_name
100	Greatest Hits	Queen
101	Greatest Hits	Boston
102	Rumours	Fleetwood Mac
103	Second Helping	Lynyrd Skynyrd

With 'varchar' specify maximum # of characters

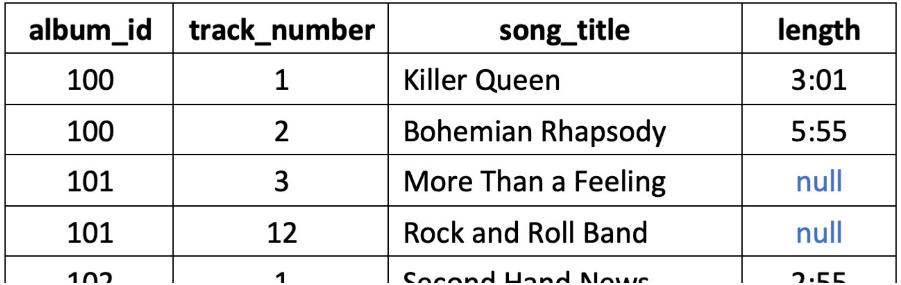
Enable the database to check for invalid values

- unsigned
- not null

Copy & paste into MariaDB from your text editor

```
create table album
(   album_id     int unsigned not null primary key,
    album_name   varchar(20) not null,
    artist_name  varchar(20) not null
);

-- Verify the table was created
show tables;
```

Creating a Second Table



track

album_id	track_number	song_title	length
100	1	Killer Queen	3:01
100	2	Bohemian Rhapsody	5:55
101	3	More Than a Feeling	null
101	12	Rock and Roll Band	null
102	1	Second Hand News	2:55

Must create 'album' first due to the FK

The 'length' column permits null values

Multi-column primary key

- So we must define the PK differently

```
create table track
(  album_id       int unsigned not null,
   track_number   int unsigned not null,
   song_title     varchar(30) not null,
   length         varchar(7),
   primary key (album_id, track_number),
   foreign key (album_id)
       references album (album_id)
);
```

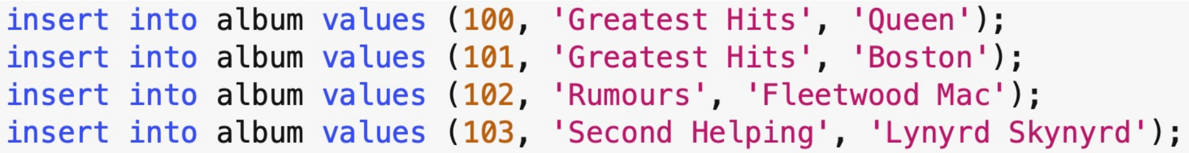
Let's create some data!

album

album_id	album_name	artist_name
100	Greatest Hits	Queen
101	Greatest Hits	Boston
102	Rumours	Fleetwood Mac
103	Second Helping	Lynyrd Skynyrd

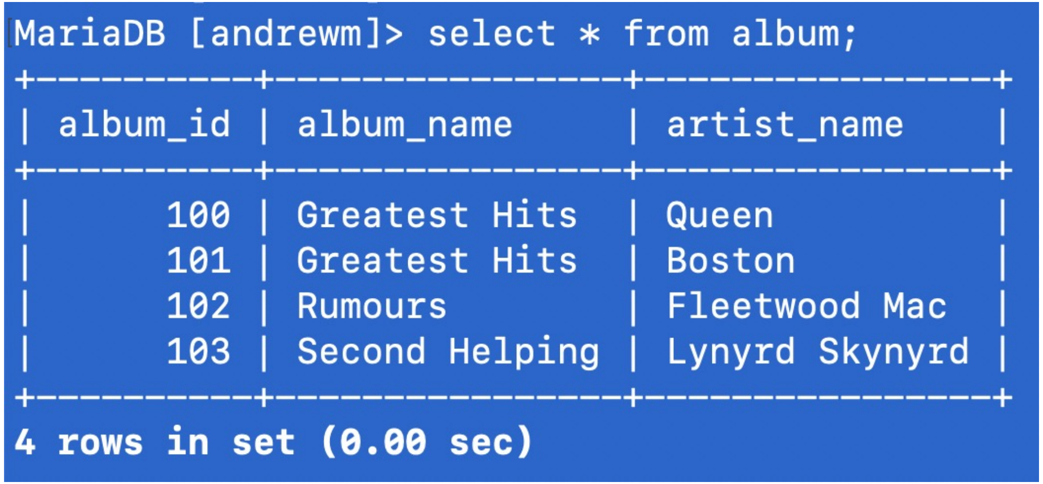
Values must:

- Be in the same order in which the columns were defined
- Have the correct data type
- Conform to PK and FK constraints



```
insert into album values (100, 'Greatest Hits', 'Queen');
insert into album values (101, 'Greatest Hits', 'Boston');
insert into album values (102, 'Rumours', 'Fleetwood Mac');
insert into album values (103, 'Second Helping', 'Lynyrd Skynyrd');
```

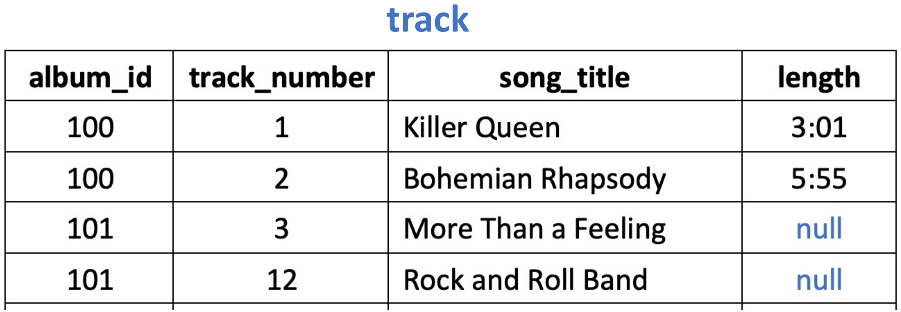
A First Select Statement



```
MariaDB [andrewm]> select * from album;
+----------+----------------+----------------+
| album_id | album_name     | artist_name    |
+----------+----------------+----------------+
|      100 | Greatest Hits  | Queen          |
|      101 | Greatest Hits  | Boston         |
|      102 | Rumours        | Fleetwood Mac  |
|      103 | Second Helping | Lynyrd Skynyrd |
+----------+----------------+----------------+
4 rows in set (0.00 sec)
```

- The simplest select statement
- All columns & rows for one table

And now some 'track' data...



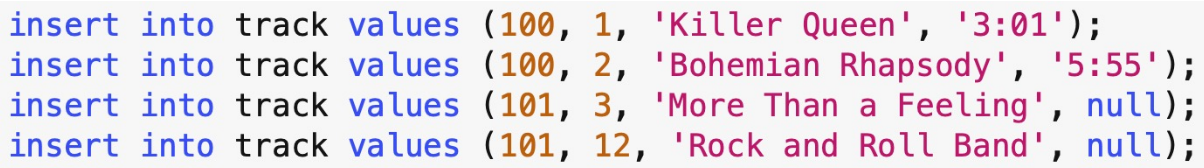
track

album_id	track_number	song_title	length
100	1	Killer Queen	3:01
100	2	Bohemian Rhapsody	5:55
101	3	More Than a Feeling	null
101	12	Rock and Roll Band	null

The 'length' column permits null values

Multi-column primary key

- So we must define the PK differently



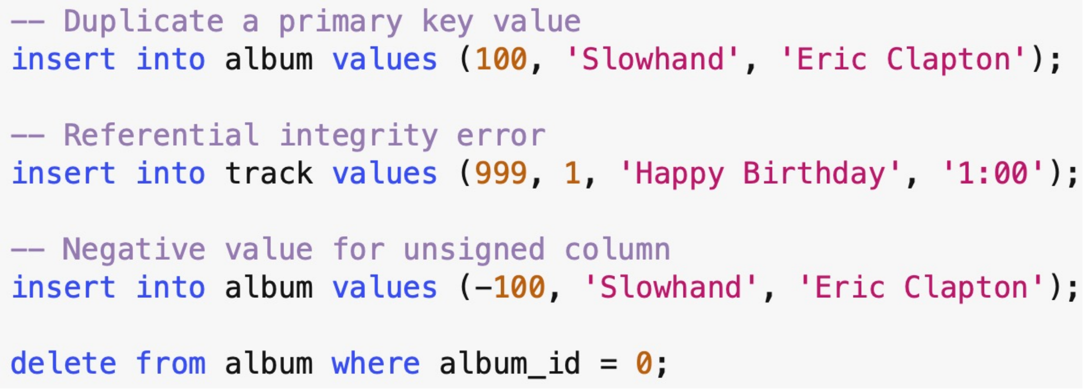
```
insert into track values (100, 1, 'Killer Queen', '3:01');
insert into track values (100, 2, 'Bohemian Rhapsody', '5:55');
insert into track values (101, 3, 'More Than a Feeling', null);
insert into track values (101, 12, 'Rock and Roll Band', null);
```

```
select * from track;
```

Insertions with Errors

album

album_id	album_name	artist_name
100	Greatest Hits	Queen
101	Greatest Hits	Boston
102	Rumours	Fleetwood Mac
103	Second Helping	Lynyrd Skynyrd

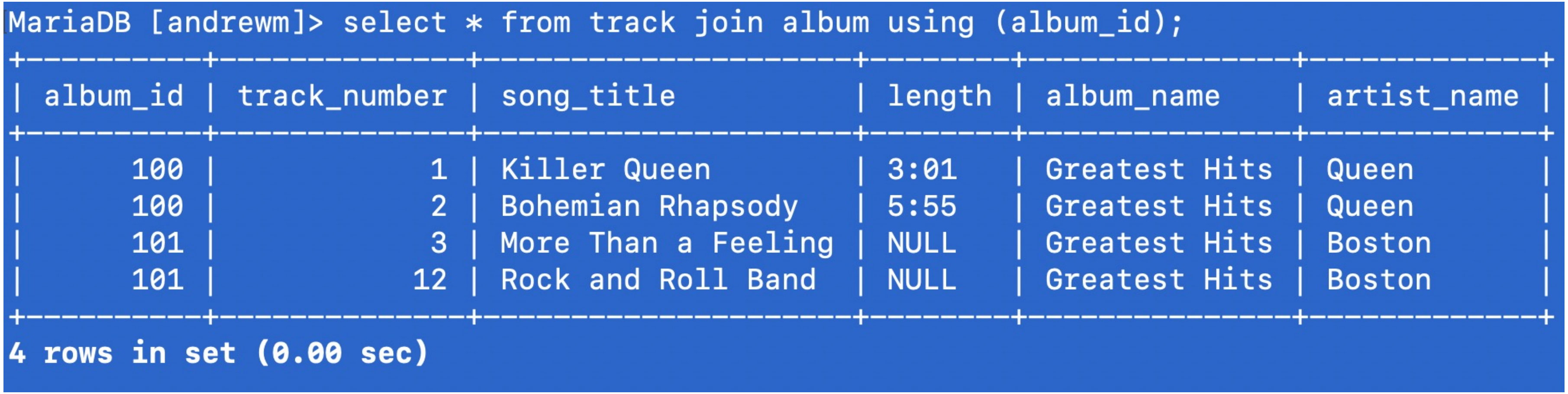


```sql
-- Duplicate a primary key value
insert into album values (100, 'Slowhand', 'Eric Clapton');

-- Referential integrity error
insert into track values (999, 1, 'Happy Birthday', '1:00');

-- Negative value for unsigned column
insert into album values (-100, 'Slowhand', 'Eric Clapton');

delete from album where album_id = 0;
```

track

album_id	track_number	song_title	length
100	1	Killer Queen	3:01
100	2	Bohemian Rhapsody	5:55
101	3	More Than a Feeling	null
101	12	Rock and Roll Band	null

Data from different tables - join

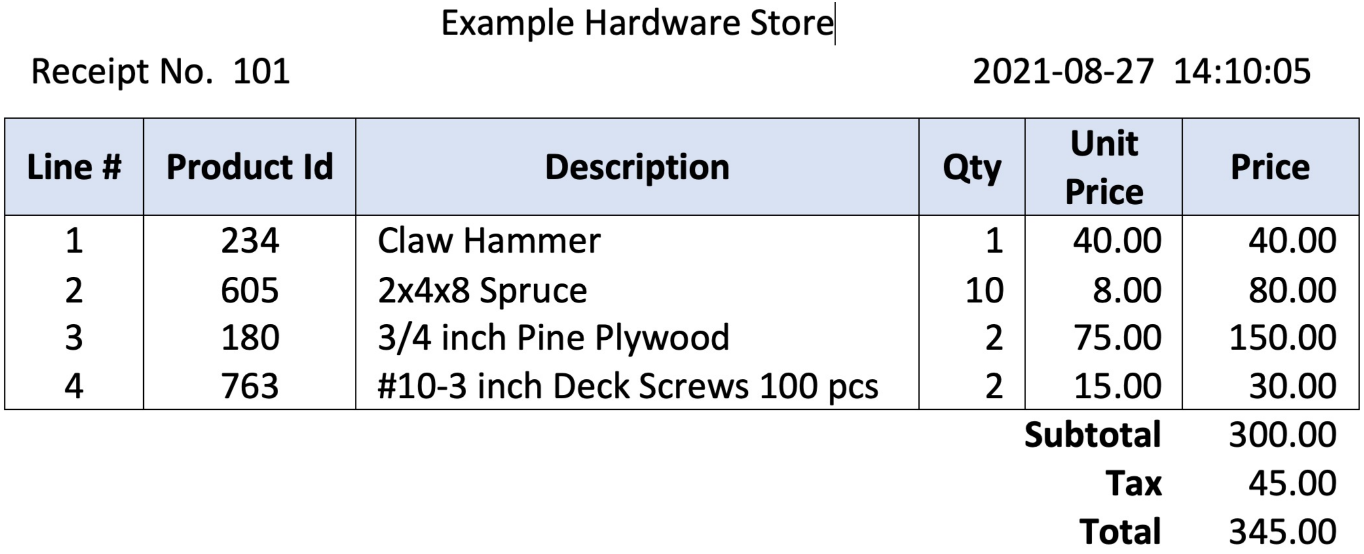


```
MariaDB [andrewm]> select * from track join album using (album_id);
+----------+--------------+---------------------+--------+---------------+-------------+
| album_id | track_number | song_title          | length | album_name    | artist_name |
+----------+--------------+---------------------+--------+---------------+-------------+
|      100 |            1 | Killer Queen        | 3:01   | Greatest Hits | Queen       |
|      100 |            2 | Bohemian Rhapsody   | 5:55   | Greatest Hits | Queen       |
|      101 |            3 | More Than a Feeling | NULL   | Greatest Hits | Boston      |
|      101 |           12 | Rock and Roll Band  | NULL   | Greatest Hits | Boston      |
+----------+--------------+---------------------+--------+---------------+-------------+
4 rows in set (0.00 sec)
```

Brings together the columns from both tables

<table1> join <table2> using (<common_field>)

Reminder - Sales Receipt Example



Example Hardware Store

Receipt No. 101 2021-08-27 14:10:05

Line #	Product Id	Description	Qty	Unit Price	Price
1	234	Claw Hammer	1	40.00	40.00
2	605	2x4x8 Spruce	10	8.00	80.00
3	180	3/4 inch Pine Plywood	2	75.00	150.00
4	763	#10-3 inch Deck Screws 100 pcs	2	15.00	30.00
				Subtotal	300.00
				Tax	45.00
				Total	345.00

Reminder – Tables for Receipts

receipt

receipt_num	date_time
101	2021-08-27 14:10:05
102	2021-08-27 14:17:43

product

prod_id	description	unit
234	Claw Hammer	40.00
605	2x4x8 Spruce	8.00
180	3/4 inch Pine Plywood	75.00
763	#10-3 inch Deck Screws 100 pcs	15.00
501	Eggshell Interior Latex Paint gal.	80.00

line_item

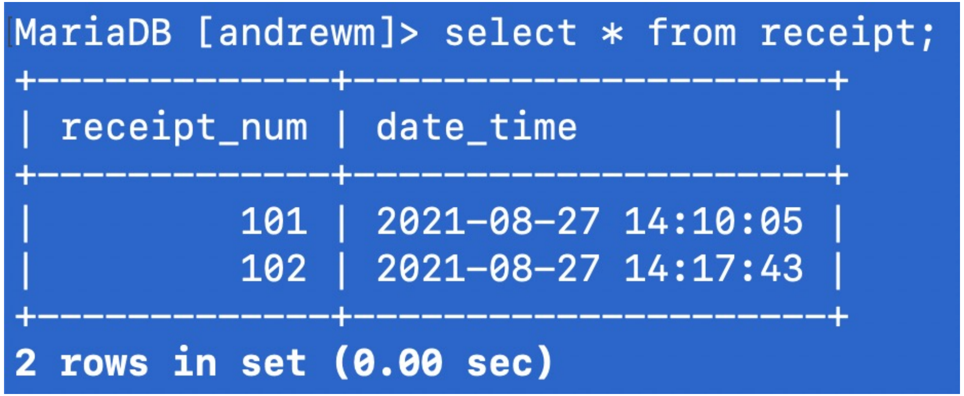
receipt_num	line	prod_id	qty
101	1	234	1
101	2	605	10
101	3	180	2
101	4	763	2
102	1	501	2
102	2	234	1

Creating the 'receipt' Table

```
create table receipt
(   receipt_num   int unsigned not null primary key,
    date_time     datetime not null
);

insert into receipt values (101, '2021-08-27 14:10:05');
insert into receipt values (102, '2021-08-27 14:17:43');
```

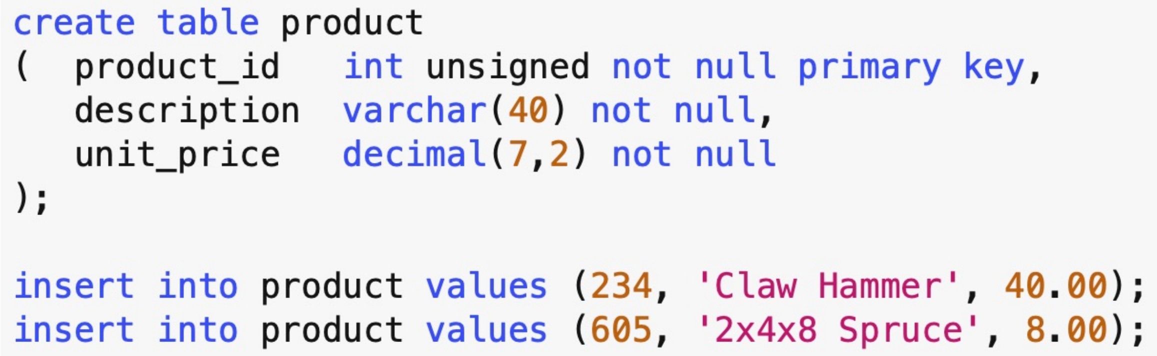
datetime values are NOT strings

- Can be compared
- Related data types:
  - date
  - time



```
MariaDB [andrewm]> select * from receipt;
+-------------+---------------------+
| receipt_num | date_time           |
+-------------+---------------------+
|         101 | 2021-08-27 14:10:05 |
|         102 | 2021-08-27 14:17:43 |
+-------------+---------------------+
2 rows in set (0.00 sec)
```

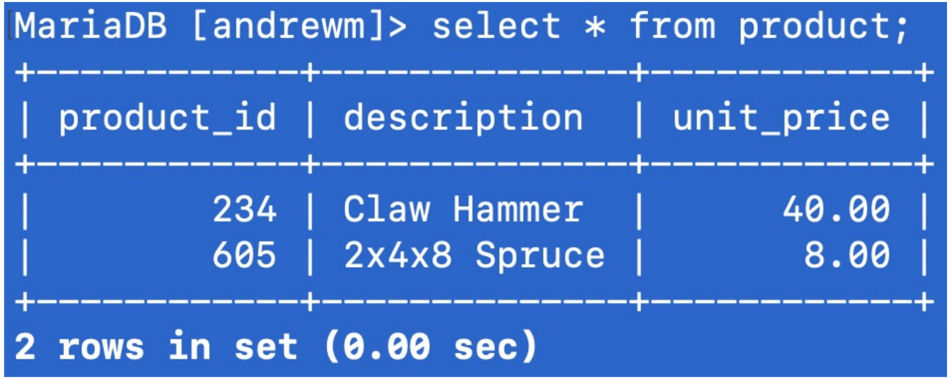
The 'product' Table



```
create table product
(   product_id   int unsigned not null primary key,
    description  varchar(40) not null,
    unit_price   decimal(7,2) not null
);

insert into product values (234, 'Claw Hammer', 40.00);
insert into product values (605, '2x4x8 Spruce', 8.00);
```

decimal is recommended for $ values

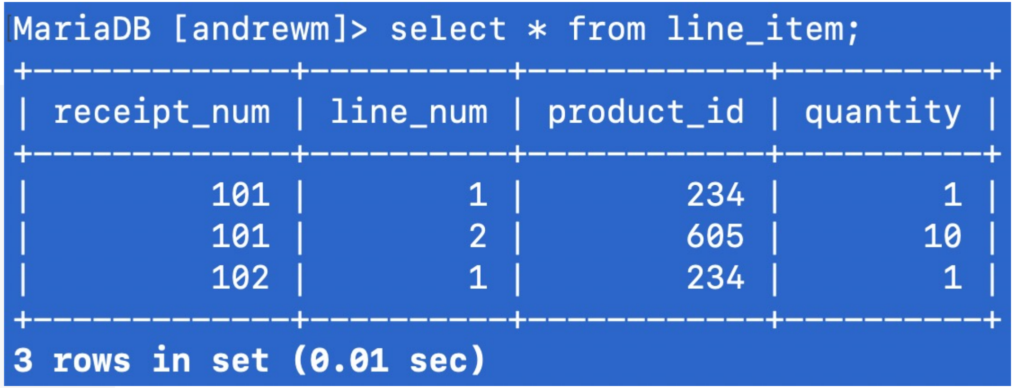
- Each digit is stored separately
- 7 total digits
- 2 digits right of the decimal point
- $0.00 to $99,999.99



```
MariaDB [andrewm]> select * from product;
+------------+--------------+------------+
| product_id | description  | unit_price |
+------------+--------------+------------+
|        234 | Claw Hammer  |      40.00 |
|        605 | 2x4x8 Spruce |       8.00 |
+------------+--------------+------------+
2 rows in set (0.00 sec)
```

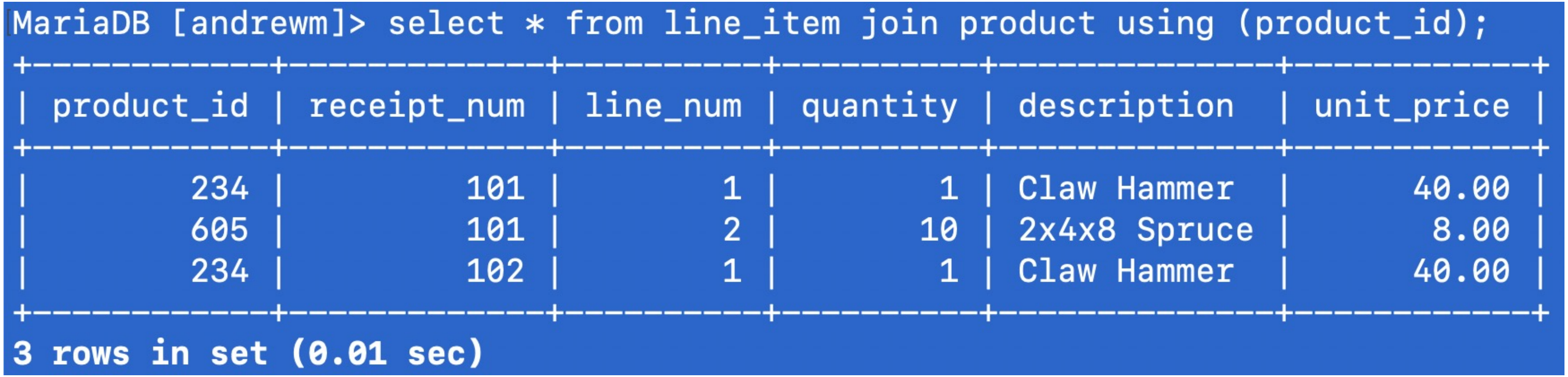
The 'line_item' Table

```sql
create table line_item
(   receipt_num  int unsigned not null,
    line_num     int unsigned not null,
    product_id   int unsigned not null,
    quantity     int unsigned not null,
    primary key (receipt_num, line_num),
    foreign key (receipt_num)
        references receipt (receipt_num),
    foreign key (product_id)
        references product (product_id)
);

insert into line_item values(101, 1, 234,  1);
insert into line_item values(101, 2, 605, 10);
insert into line_item values(102, 1, 234,  1);
```



```
MariaDB [andrewm]> select * from line_item;
+-------------+----------+------------+----------+
| receipt_num | line_num | product_id | quantity |
+-------------+----------+------------+----------+
|         101 |        1 |        234 |        1 |
|         101 |        2 |        605 |       10 |
|         102 |        1 |        234 |        1 |
+-------------+----------+------------+----------+
3 rows in set (0.01 sec)
```

Displaying product data with line items



```
MariaDB [andrewm]> select * from line_item join product using (product_id);
+------------+-------------+----------+----------+--------------+------------+
| product_id | receipt_num | line_num | quantity | description  | unit_price |
+------------+-------------+----------+----------+--------------+------------+
|        234 |         101 |        1 |        1 | Claw Hammer  |      40.00 |
|        605 |         101 |        2 |       10 | 2x4x8 Spruce |       8.00 |
|        234 |         102 |        1 |        1 | Claw Hammer  |      40.00 |
+------------+-------------+----------+----------+--------------+------------+
3 rows in set (0.01 sec)
```

The End

☺