

## Beermaster: Here it Comes Again

Code I wrote for the loads

Loadbar

```
CREATE DEFINER='root'@'localhost' PROCEDURE `loadbar`()
```

```
BEGIN
```

```
declare v_finished varchar(45);
```

```
declare v_idlocations varchar(45);
```

```
declare v_name varchar(45);
```

```
DECLARE v_address varchar(45);
```

```
DECLARE v_status varchar(10);
```

```
DECLARE customer_cur CURSOR FOR
```

```
select idlocations, name, concat(concat(address,concat(' ',concat(city,concat(' ',concat(state, ' ',zipcode)) AS address from beermaster2.templocations;
```

```
-- declare NOT FOUND handler
```

```
DECLARE CONTINUE HANDLER
```

```
FOR NOT FOUND SET v_finished = 1;
```

```
-- get records from IOC
```

```
OPEN customer_cur;
```

```
get_customer: LOOP
```

```
FETCH customer_cur into
```

```
v_idlocations,
```

```
v_name,
```

```
v_address;
```

```
IF v_finished = 1 THEN
```

```
LEAVE get_customer;
```

```
END IF;
```

```
INSERT INTO beermaster2.bar
```

```
(
```

```
  `bar name`,
```

```
  `address`,
```

```
  `barid`
```

```
)
```

```
VALUES
```

```
(
```

```
  v_name,
```

```
  v_address,
```

```
  v_idlocations
```

```
);
```

```
END LOOP get_customer;
```

```
CLOSE customer_cur;
```

```
END
```

```
Loadbeer
```

```
CREATE DEFINER='root'@'localhost' PROCEDURE `loadBeer`()
```

```
BEGIN
```

```
declare v_finished varchar(45);
```

```
declare v_Brandid varchar(45);
```

```
DECLARE v_brandname varchar(45);
```

```
DECLARE v_Breweryname varchar(45);
```

```
DECLARE v_status varchar(10);
```

```
DECLARE customer_cur CURSOR FOR
```

```
select Brandid, brandname,Breweryname from beermaster2.tempbeers;
```

```
-- declare NOT FOUND handler
```

```
DECLARE CONTINUE HANDLER
```

```
FOR NOT FOUND SET v_finished = 1;
```

```
-- get records from IOC
```

```
OPEN customer_cur;
```

```
get_customer: LOOP
```

```
FETCH customer_cur into
```

```
v_Brandid,
```

```
v_brandname,
```

```
v_Breweryname;
```

```
IF v_finished = 1 THEN
```

```
LEAVE get_customer;
```

```
END IF;
```

```
INSERT INTO beermaster2.beers
```

```
(
```

```
Brandid`,
```

```
`name`,
```

```
`manf`
```

```

        )
        VALUES
        (
        v_Brandid,
            v_brandname,
        v_Breweryname);
    END LOOP get_customer;

CLOSE
customer_cur;

END

Loadrinker code

CREATE DEFINER=`root`@`localhost` PROCEDURE `loadDrinker`()
BEGIN
    declare v_finished varchar(45);
    declare v_customerid varchar(45);
    DECLARE v_customer varchar(100);
    DECLARE v_status varchar(10);
    DECLARE customer_cur CURSOR FOR
    select concat(concat(first, ' '),last) AS Customer, customerid from beermaster2.temptable;

    -- declare NOT FOUND handler
    DECLARE CONTINUE HANDLER
    FOR NOT FOUND SET v_finished = 1;

    -- get records from IOC
    OPEN customer_cur;
    get_customer: LOOP

```

```
FETCH customer_cur into  
v_customerid,  
v_customer;
```

```
IF v_finished = 1 THEN  
LEAVE get_customer;  
END IF;
```

```
INSERT INTO beermaster2.drinkers  
(  
`customerid`,  
`name`  
)  
VALUES  
(  
v_customer,  
v_customerid);
```

```
END LOOP get_customer;
```

```
CLOSE customer_cur;  
END
```

## Display Temp beers

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'beermeister2' selected, showing tables like 'beer', 'beers', 'beers2', 'beers3', 'beers4', 'beers5', 'beers6', 'beers7', 'beers8', 'beers9', 'beers10', 'beers11', 'beers12', 'beers13', 'beers14', 'beers15', 'beers16', 'beers17', 'beers18', 'beers19', 'beers20', 'beers21', 'beers22', 'beers23', 'beers24', 'beers25', 'beers26', 'beers27', 'beers28', 'beers29', 'beers30', 'beers31', 'beers32', 'beers33', 'beers34', 'beers35', 'beers36', 'beers37', 'beers38', 'beers39', 'beers40', 'beers41', 'beers42', 'beers43', 'beers44', 'beers45', 'beers46', 'beers47', 'beers48', 'beers49', 'beers50', 'beers51', 'beers52', 'beers53', 'beers54', 'beers55', 'beers56', 'beers57', 'beers58', 'beers59', 'beers60', 'beers61', 'beers62', 'beers63', 'beers64', 'beers65', 'beers66', 'beers67', 'beers68', 'beers69', 'beers70', 'beers71', 'beers72', 'beers73', 'beers74', 'beers75', 'beers76', 'beers77', 'beers78', 'beers79', 'beers80', 'beers81', 'beers82', 'beers83', 'beers84', 'beers85', 'beers86', 'beers87', 'beers88', 'beers89', 'beers90', 'beers91', 'beers92', 'beers93', 'beers94', 'beers95', 'beers96', 'beers97', 'beers98', 'beers99', 'beers100'. The main query editor shows the query: `SELECT * FROM beermeister2.tempbeers;`. The results pane displays a table with columns: brandid, brandname, brewerid, brewername, city, state, country, and description. The output pane shows the execution time and message: `17:28:47 SELECT * FROM beermeister2.tempbeers LIMIT 0, 1000 24 rows returned`.

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

brandid	brandname	brewerid	brewername	city	state	country	Description
10001	A&W	1819	Dr Pepper Snapple Group	Lodi	CA	United States	After writing up the first /
10002	A.J. Stephens	1835	A.J. Stephens Beverages	Pull River	MA	United States	A.J. Stephens Company ma
10003	Abita	1986	Abita Brewery	Covington	LA	United States	Abita Root Beer is made w
10004	Bayer	1896	Coca-Cola	New Orleans	LA	United States	Since 1896 Bayer's root be
10005	Bedford	1894	NamPent Soda Works	Port Angeles	WA	United States	Alvins Ice Cold, "Preme Int
10006	Bulldog	1997	Bulldog Brewing Company	Pasadena	CA	United States	Bulldog Root Beer is differ
10007	Bundaberg	1940	Bundaberg Brewed Drinks	Bundaberg	Qld	Australia	Savour the taste of the p...

Table: tempbeers

Columns:

- brandid: int(11)
- brandname: varchar(45)
- brewerid: varchar(45)
- brewername: varchar(45)
- city: varchar(45)
- state: varchar(45)
- country: varchar(45)
- Description: varchar(1000)
- canesugar: varchar(45)
- canesugar2: varchar(45)
- honey: varchar(45)
- artificial sweetener: varchar(45)
- caffeinated: varchar(45)
- alcoholic: varchar(45)
- available in cans: varchar(45)
- available in bottles: varchar(45)

Action Output

Time	Action	Message	Duration / Fetch
17:28:47	SELECT * FROM beermeister2.tempbeers LIMIT 0, 1000	24 rows returned	0.000 sec / 0.000 sec

## Display templocations(bar)

The screenshot shows the MySQL Workbench interface. The left sidebar displays the database schema, with the **templocations** table selected under the **beermeister2** database. The main query editor contains the SQL statement: `SELECT * FROM beermeister2.templocations;`. The results are displayed in a table with the following data:

idlocations	name	address	city	state	zipcode
1	Sec State American River Courtyard	6000 J St	Secramento	CA	95818
2	Sec State Union	6000 J St	Secramento	CA	95818
3	Her to Brewery	101 3500 oakland	1000 oak	CA	91360
4	Big A's	123 Gwy St	Laad	CA	91324

Below the results, the 'Table: templocations' section lists the columns and their data types:

Column	Data Type
idlocations	int(11)
name	varchar(45)
address	varchar(45)
city	varchar(45)
state	varchar(45)
zipcode	varchar(45)

The 'Action Output' section shows the execution results of the query:

Time	Action	Message	Duration / Rows
17:28:47	SELECT * FROM beermeister2.templocations LIMIT 0, 1000	24 rows returned	0.006 sec / 0.000 sec
17:30:39	SELECT * FROM beermeister2.templocations LIMIT 0, 1000	4 rows returned	0.016 sec / 0.000 sec

## Temptable(drinkers)

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'beermeister2' expanded, showing tables like 'beerjournal', 'beers', 'drinks', 'likes', 'sells', 'tempbeers', 'templocations', and 'temptable'. The central query editor contains the SQL statement: `SELECT * FROM beermeister2.temptable;`. The 'Results Grid' below shows a table with 9 columns: customerid, first, last, street address, city, state, zipcode, email, and phone number. It contains 6 rows of data. The bottom panel shows the 'Action Output' with a log of three SQL statements executed, each limited to 1000 rows. The first statement returned 24 rows, the second returned 4 rows, and the third returned 554 rows.

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

customerid	first	last	street address	city	state	zipcode	email	phone number
113811	Nemeth	Walter	6715 Commonwealth Dr	Sacramento	CA	94236	walter.x.76@fastmail.com	(916) 918-1561
113368	Maddone	Jonas	3603 Lodi Way	Sacramento	CA	94238	j_maddone@gmail.com	(916) 286-9423
114938	Ganes	Wheeler	4740 Ironwood Way	Orangevale	CA	95662	dhwheeler@outlook.com	(916) 364-1156
110546	Kevin	Gilbert	3188 Livingston Way	Polson	CA	95671	kgilbert@fastmail.com	(916) 304-9859
115771	John	Young	663 Blackhawk Way	Sacramento	CA	95830	john.y90@gmail.com	(916) 738-4129
118708	Tummy	Kara	3179 Canada River Way	Yuba City	CA	95691	tkara@outlook.com	(916) 115-2967
119451	Anna	Hines	5917 Chicago Way	Sacramento	CA	94207	h_anna@fastmail.com	(916) 399-9669

Table: temptable

Columns:

- customerid: int(11) (PK)
- first: varchar(45)
- last: varchar(45)
- street address: varchar(45)
- city: varchar(45)
- state: varchar(45)
- zipcode: varchar(45)
- email: varchar(45)
- phone number: varchar(45)
- first purchase date: varchar(45)
- authorized to email let: varchar(45)
- Ganzler: varchar(45)

Action Output

Time	Action	Message	Duration / Rows
17:28:47	SELECT * FROM beermeister2.tempbeers LIMIT 0, 1000	24 rows returned	0.000 sec / 0.000 sec
17:30:09	SELECT * FROM beermeister2.templocations LIMIT 0, 1000	4 rows returned	0.016 sec / 0.000 sec
17:31:14	SELECT * FROM beermeister2.temptable LIMIT 0, 1000	554 rows returned	0.015 sec / 0.000 sec

## Loadbar result





## Load beer result

The screenshot displays the MySQL Workbench interface during a load operation. The left sidebar shows the database schema, with the 'beers' table selected under the 'beermaster2' database. The main window shows the SQL editor with the query `SELECT * FROM beermaster2.beers;`. Below the editor, the 'Result Grid' displays a list of beer records. The bottom panel shows the 'Action Output' window, which provides a detailed log of the load operation, including the time, action, message, and duration for each step.

**Table: beers**

id	name	manf	container	top	brandid
1	Karl's Killer Cola	Rad Man	Can	Can top	0001
2	Beer	Rad Man	Can	Can	0002
3	Asahi	Onyiah Shippie Group	Can	Cap	0003
4	A.L. Stephens	A.L. Stephens Beverages	Can	Cap	0004
5	Abita	Abita Brewery	Can	Cap	0005
6	Beck's	Coca-Cola	Can	Cap	0006
7	Bedford	Northwest Soda Works	Can	Cap	0007
8	Budweiser	Budweiser Brewing Company	Can	Cap	0008
9	Bundberg	Bundberg Brewed Drinks	Can	Cap	0009

**Action Output**

#	Time	Action	Message	Duration / Fetch
6	17:41:43	call beermaster2.loadBeer()	0 rows affected	0.031 sec
7	17:41:51	SELECT * FROM beermaster2.beer LIMIT 0, 1000	14 rows returned	0.000 sec / 0.000 sec
8	17:44:43	SELECT * FROM beermaster2.beers LIMIT 0, 1000	25 rows returned	0.000 sec / 0.000 sec
9	17:45:18	SELECT * FROM beermaster2.beer LIMIT 0, 1000	14 rows returned	0.000 sec / 0.000 sec
10	17:45:47	SELECT * FROM beermaster2.beers LIMIT 0, 1000	25 rows returned	0.000 sec / 0.000 sec
11	17:46:22	call beermaster2.loadBeer()	0 rows affected	0.141 sec
12	17:46:34	call beermaster2.loadBeer()	0 rows affected	0.157 sec
13	17:46:35	call beermaster2.loadBeer()	0 rows affected	0.130 sec
14	17:46:34	SELECT * FROM beermaster2.beers LIMIT 0, 1000	74 rows returned	0.000 sec / 0.000 sec
15	17:47:17	SELECT * FROM beermaster2.beers LIMIT 0, 1000	74 rows returned	0.000 sec / 0.016 sec

## Loaddrinkers

The screenshot shows the MySQL Workbench interface with a query executed in the 'SQL Editor' tab. The query is `SELECT * FROM beermaster2.drinkers;`. The 'Result Grid' displays a list of customers with their IDs and names. The 'Action Output' pane shows the execution details of the query, including the time taken and the number of rows returned.

**Table: drinkers**

id	name
1	Jack
2	John
3891	Kenneth Wilson
3892	Madeline Jones
3893	Carson Wheeler
3894	Kevin Gilbert
3895	John Young
3896	Timothy Horne
3897	Anna Hines

**Action Output**

#	Time	Action	Message	Duration / Batch
7	17:41:51	SELECT * FROM beermaster2.beer LIMIT 0, 1000	14 row(s) returned	0.000 sec / 0.000 sec
8	17:44:43	SELECT * FROM beermaster2.beers LIMIT 0, 1000	26 row(s) returned	0.000 sec / 0.000 sec
9	17:45:18	SELECT * FROM beermaster2.beer LIMIT 0, 1000	14 row(s) returned	0.000 sec / 0.000 sec
10	17:45:47	SELECT * FROM beermaster2.beers LIMIT 0, 1000	26 row(s) returned	0.000 sec / 0.000 sec
11	17:46:22	call beermaster2.loadBeer()	0 row(s) affected	0.141 sec
12	17:46:24	call beermaster2.loadBeer()	0 row(s) affected	0.157 sec
13	17:46:25	call beermaster2.loadBeer()	0 row(s) affected	0.109 sec
14	17:46:34	SELECT * FROM beermaster2.beers LIMIT 0, 1000	74 row(s) returned	0.000 sec / 0.000 sec
15	17:47:17	SELECT * FROM beermaster2.beers LIMIT 0, 1000	74 row(s) returned	0.000 sec / 0.016 sec
16	17:49:45	SELECT * FROM beermaster2.drinkers LIMIT 0, 1000	1000 row(s) returned	0.015 sec / 0.000 sec
17	17:51:36	SP: FCT * FROM beermaster2.drinkers LIMIT 0, 1000	117 row(s) returned	0.000 sec / 0.000 sec