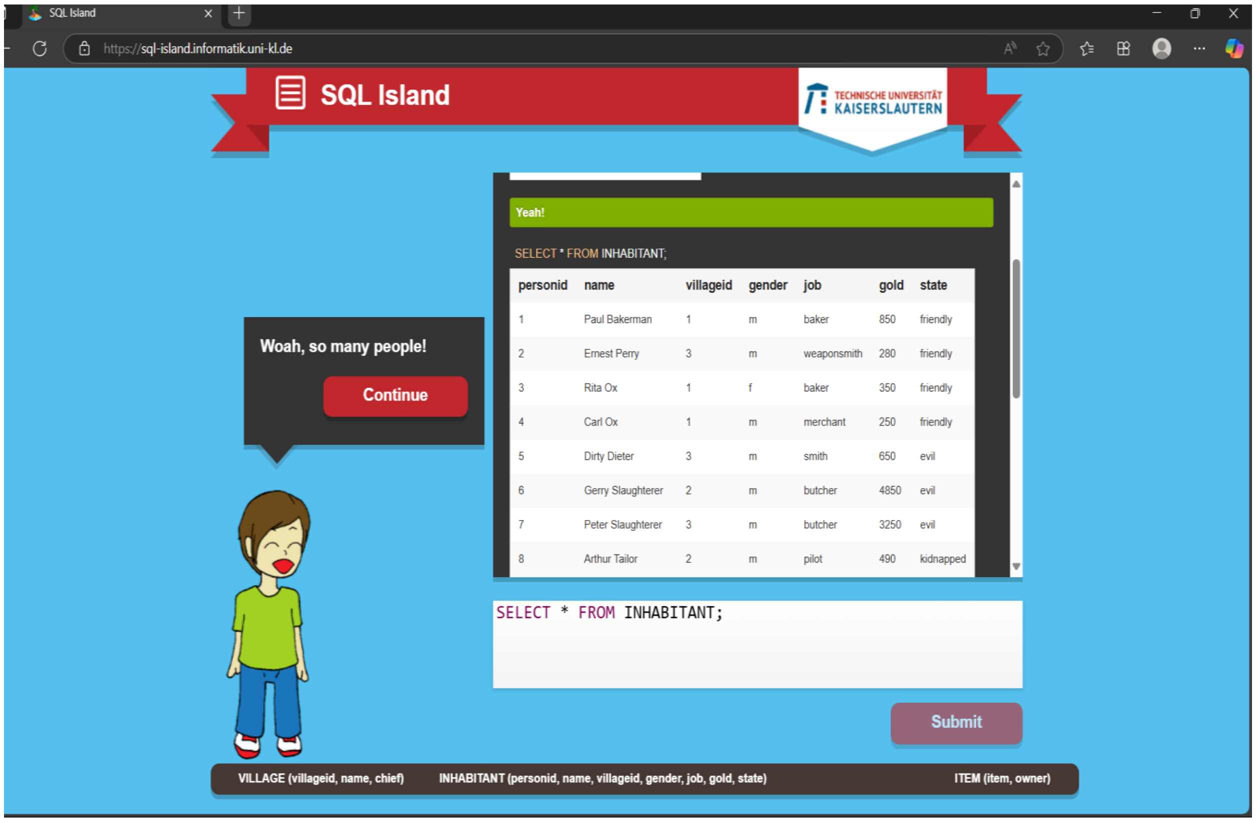
SQL ISLAND SEMI-PROJECT

Name: James Olandria Cuso Date: May 2, 2025


SQL Island
https://sql-island.informatik.uni-kl.de
SQL Island
TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
Woah, so many people!
Continue
Submit

Yeah!

SELECT * FROM INHABITANT;

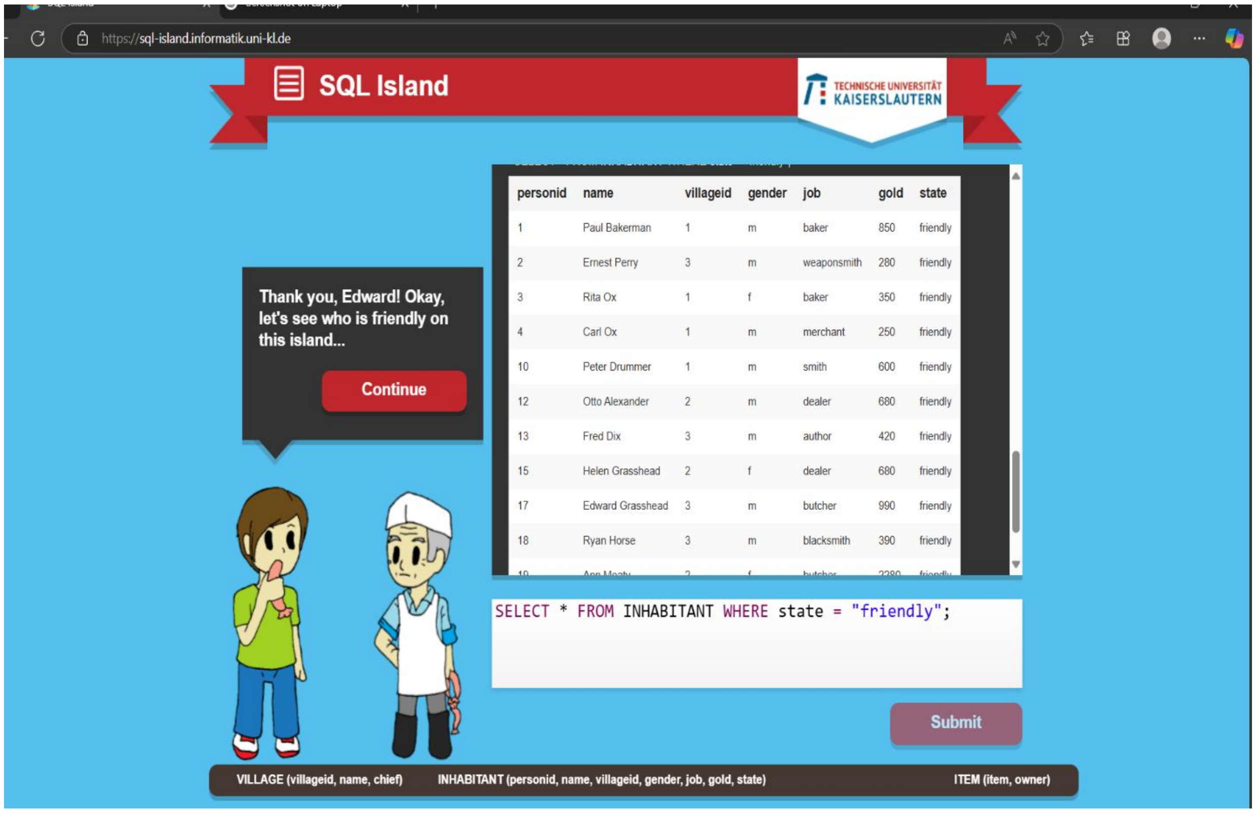
personid	name	villageid	gender	job	gold	state
1	Paul Bakerman	1	m	baker	850	friendly
2	Ernest Perry	3	m	weaponsmith	280	friendly
3	Rita Ox	1	f	baker	350	friendly
4	Carl Ox	1	m	merchant	250	friendly
5	Dirty Dieter	3	m	smith	650	evil
6	Gerry Slaughterer	2	m	butcher	4850	evil
7	Peter Slaughterer	3	m	butcher	3250	evil
8	Arthur Tailor	2	m	pilot	490	kidnapped

```
SELECT * FROM INHABITANT;
```

VILLAGE (villageid, name, chief) INHABITANT (personid, name, villageid, gender, job, gold, state) ITEM (item, owner)

SELECT * FROM INHABITANT;

This will display all the inhabitants from the INHABITANT table.

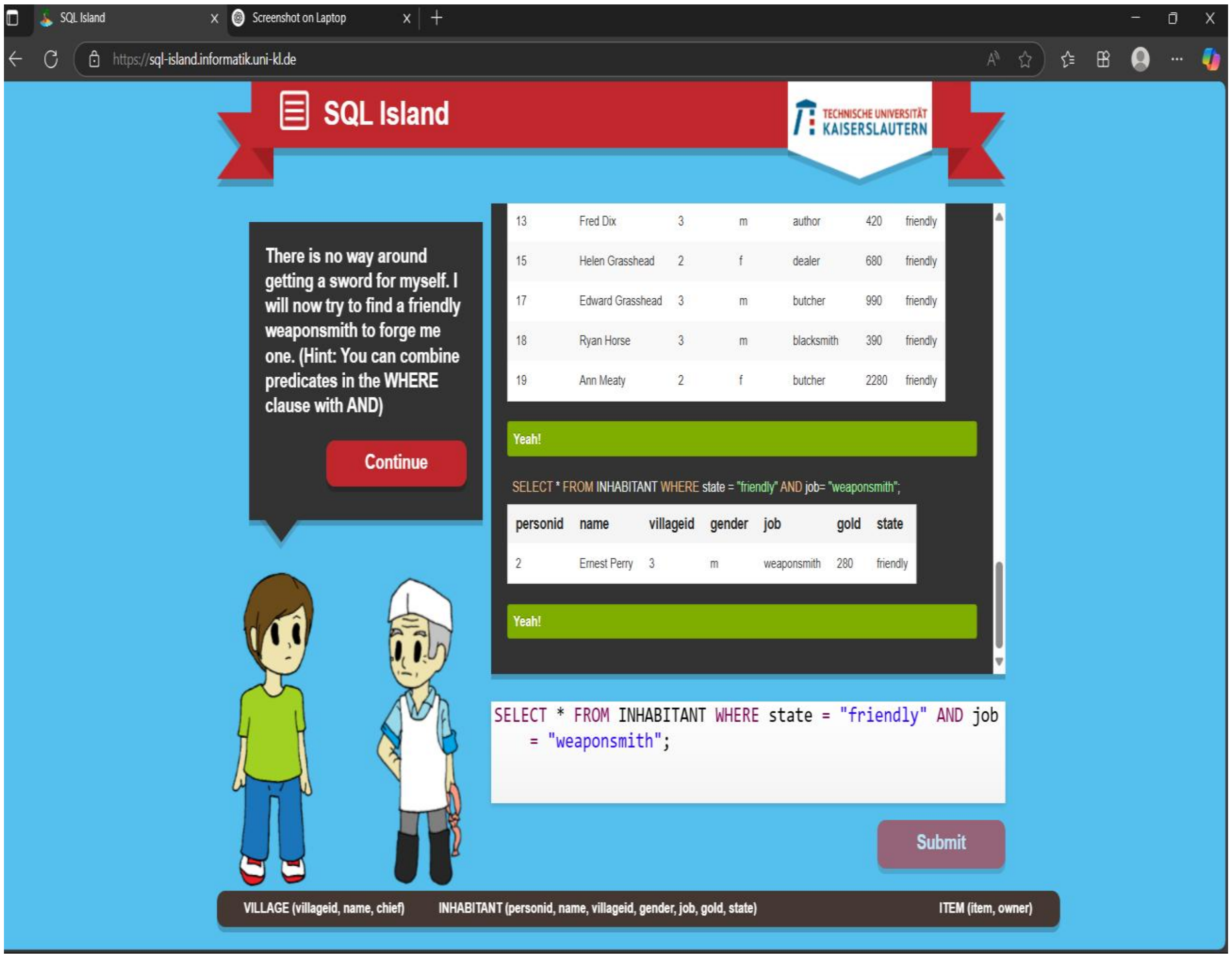

https://sql-island.informatik.uni-kl.de
SQL Island
TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
Thank you, Edward! Okay, let's see who is friendly on this island...
Continue
Submit
VILLAGE (villageid, name, chief)
INHABITANT (personid, name, villageid, gender, job, gold, state)
ITEM (item, owner)

personid	name	villageid	gender	job	gold	state
1	Paul Bakerman	1	m	baker	850	friendly
2	Ernest Perry	3	m	weaponsmith	280	friendly
3	Rita Ox	1	f	baker	350	friendly
4	Carl Ox	1	m	merchant	250	friendly
10	Peter Drummer	1	m	smith	600	friendly
12	Otto Alexander	2	m	dealer	680	friendly
13	Fred Dix	3	m	author	420	friendly
15	Helen Grasshead	2	f	dealer	680	friendly
17	Edward Grasshead	3	m	butcher	990	friendly
18	Ryan Horse	3	m	blacksmith	390	friendly
19	Ann Meaty	2	f	butcher	2280	friendly

```
SELECT * FROM INHABITANT WHERE state = "friendly";
```

SELECT * FROM INHABITANT WHERE state='friendly';

This screenshot shows my SELECT query to display all inhabitants who are in a friendly state.



There is no way around getting a sword for myself. I will now try to find a friendly weaponsmith to forge me one. (Hint: You can combine predicates in the WHERE clause with AND)

13	Fred Dix	3	m	author	420	friendly
15	Helen Grasshead	2	f	dealer	680	friendly
17	Edward Grasshead	3	m	butcher	990	friendly
18	Ryan Horse	3	m	blacksmith	390	friendly
19	Ann Meaty	2	f	butcher	2280	friendly

Yeah!

```
SELECT * FROM INHABITANT WHERE state = "friendly" AND job= "weaponsmith";
```

personid	name	villageid	gender	job	gold	state
2	Ernest Perry	3	m	weaponsmith	280	friendly

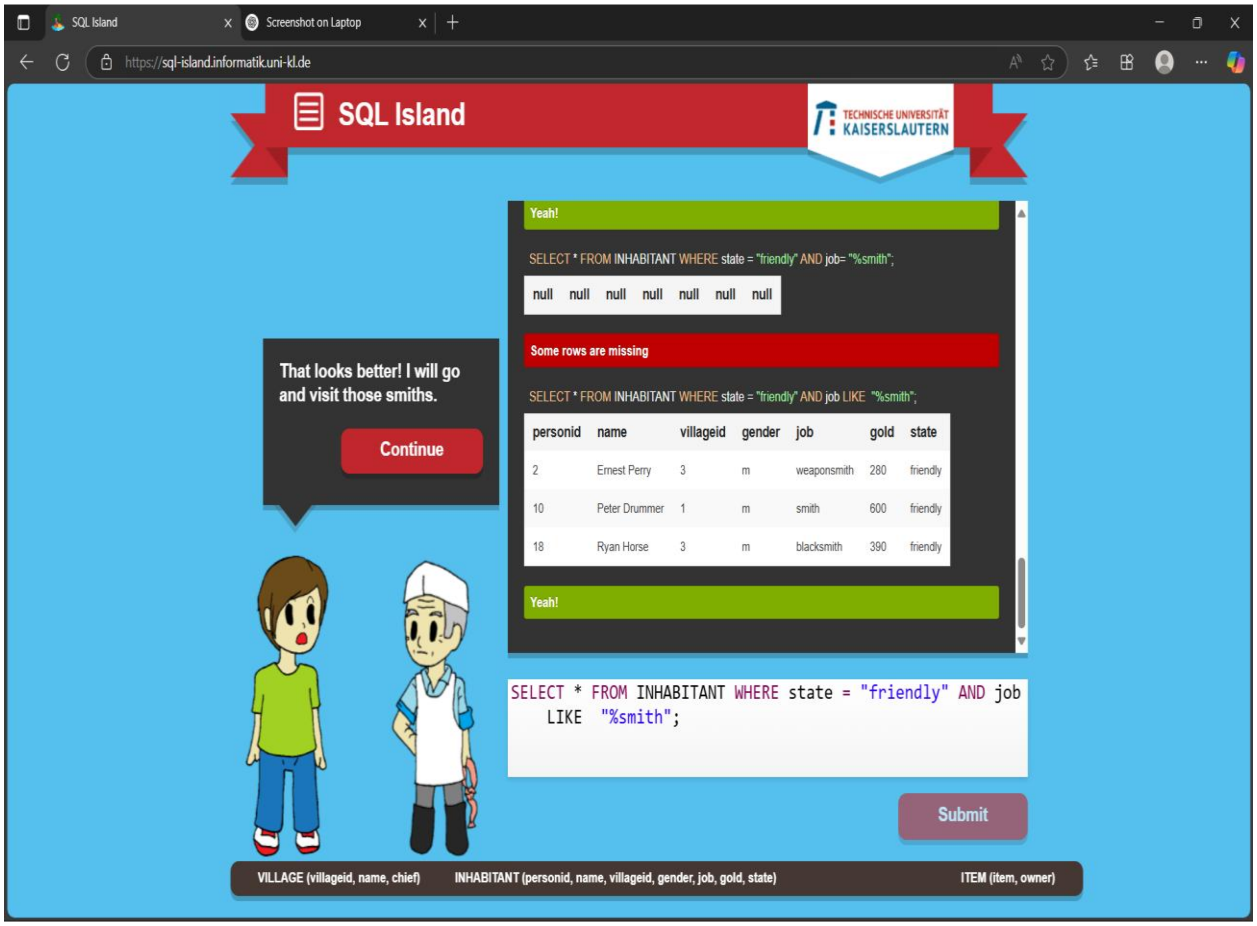
Yeah!

```
SELECT * FROM INHABITANT WHERE state = "friendly" AND job
    = "weaponsmith";
```

VILLAGE (villageid, name, chief) INHABITANT (personid, name, villageid, gender, job, gold, state) ITEM (item, owner)

SELECT * FROM INHABITANT WHERE state='friendly' AND job = 'weaponsmith';

This screenshot shows my query to find friendly inhabitants whose job is specifically a weaponsmith.


SQL Island
Screenshot on Laptop
https://sql-island.informatik.uni-kl.de
SQL Island
TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
That looks better! I will go and visit those smiths.
Continue
Submit

Yeah!

```
SELECT * FROM INHABITANT WHERE state = "friendly" AND job= "%smith";
```

```
null null null null null null null
```

Some rows are missing

```
SELECT * FROM INHABITANT WHERE state = "friendly" AND job LIKE "%smith";
```

personid	name	villageid	gender	job	gold	state
2	Ernest Perry	3	m	weaponsmith	280	friendly
10	Peter Drummer	1	m	smith	600	friendly
18	Ryan Horse	3	m	blacksmith	390	friendly

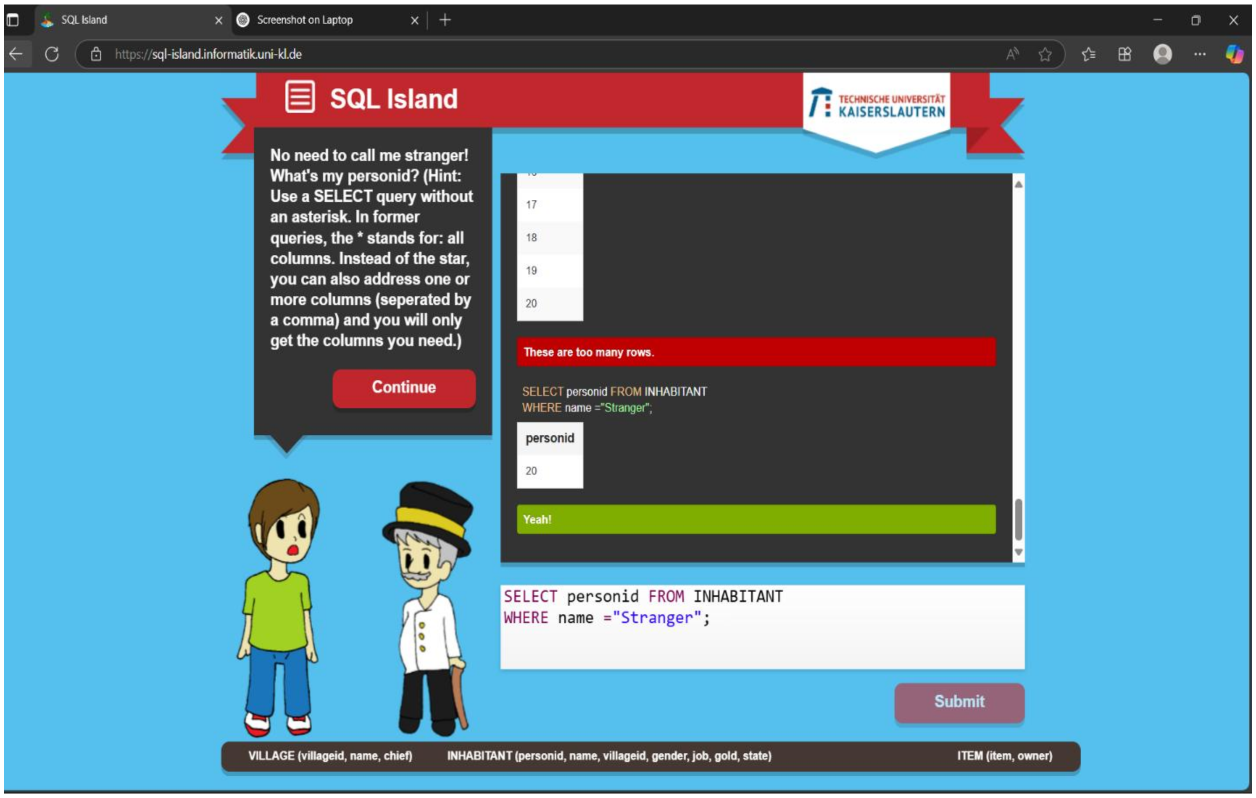
Yeah!

```
SELECT * FROM INHABITANT WHERE state = "friendly" AND job
    LIKE  "%smith";
```

VILLAGE (villageid, name, chief)   INHABITANT (personid, name, villageid, gender, job, gold, state)   ITEM (item, owner)

SELECT * FROM INHABITANT WHERE state='friendly' AND job LIKE '%smith';

This screenshot shows my query for friendly inhabitants whose job ends with 'smith' (like weaponsmith, blacksmith, etc.).


SQL Island
Screenshot on Laptop
https://sql-island.informatik.uni-kl.de
SQL Island
TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
Continue
Submit

No need to call me stranger! What's my personid? (Hint: Use a SELECT query without an asterisk. In former queries, the * stands for: all columns. Instead of the star, you can also address one or more columns (seperated by a comma) and you will only get the columns you need.)

17
18
19
20

These are too many rows.

```
SELECT personid FROM INHABITANT
WHERE name ="Stranger";
```

personid
20

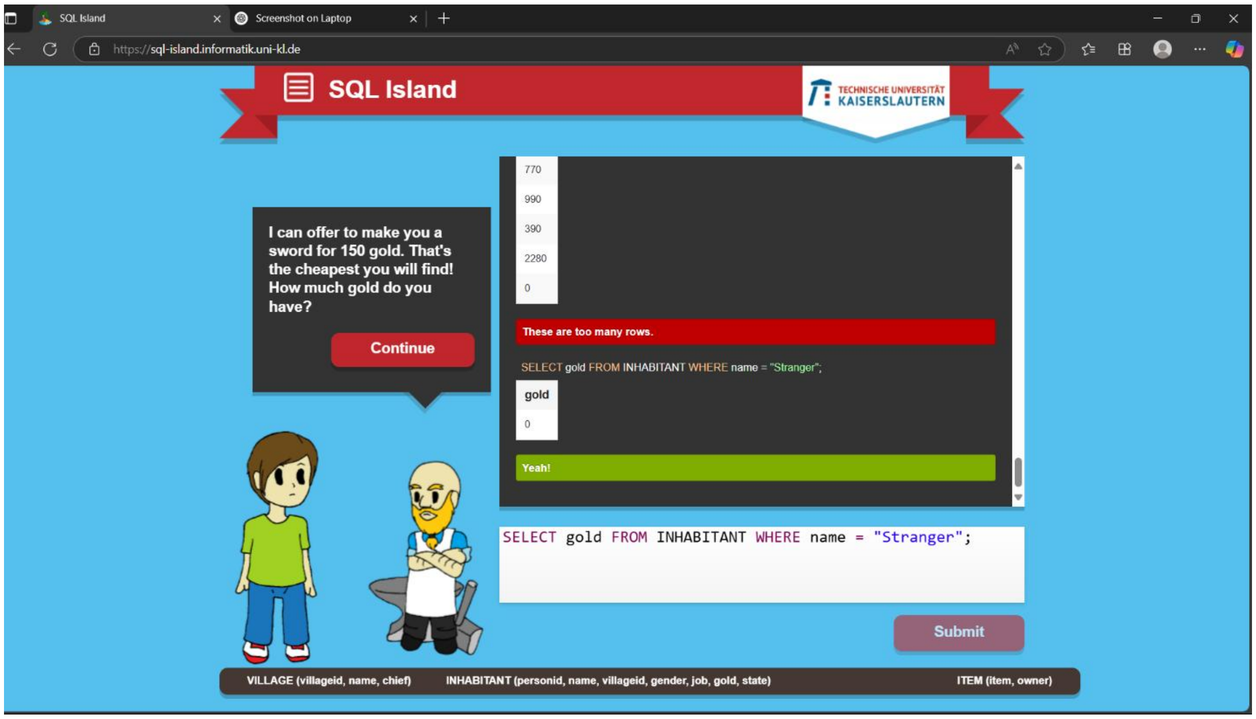
Yeah!

```
SELECT personid FROM INHABITANT
WHERE name ="Stranger";
```

VILLAGE (villageid, name, chief)   INHABITANT (personid, name, villageid, gender, job, gold, state)   ITEM (item, owner)

SELECT personid FROM INHABITANT WHERE name= 'Stranger';

This screenshot shows the query to find the person ID of the inhabitant named 'Stranger'.


SQL Island
Screenshot on Laptop
https://sql-island.informatik.uni-kl.de
SQL Island
TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
Continue
Submit
VILLAGE (villageid, name, chief)
INHABITANT (personid, name, villageid, gender, job, gold, state)
ITEM (item, owner)

I can offer to make you a sword for 150 gold. That's the cheapest you will find! How much gold do you have?

```
770
990
390
2280
0
```

These are too many rows.

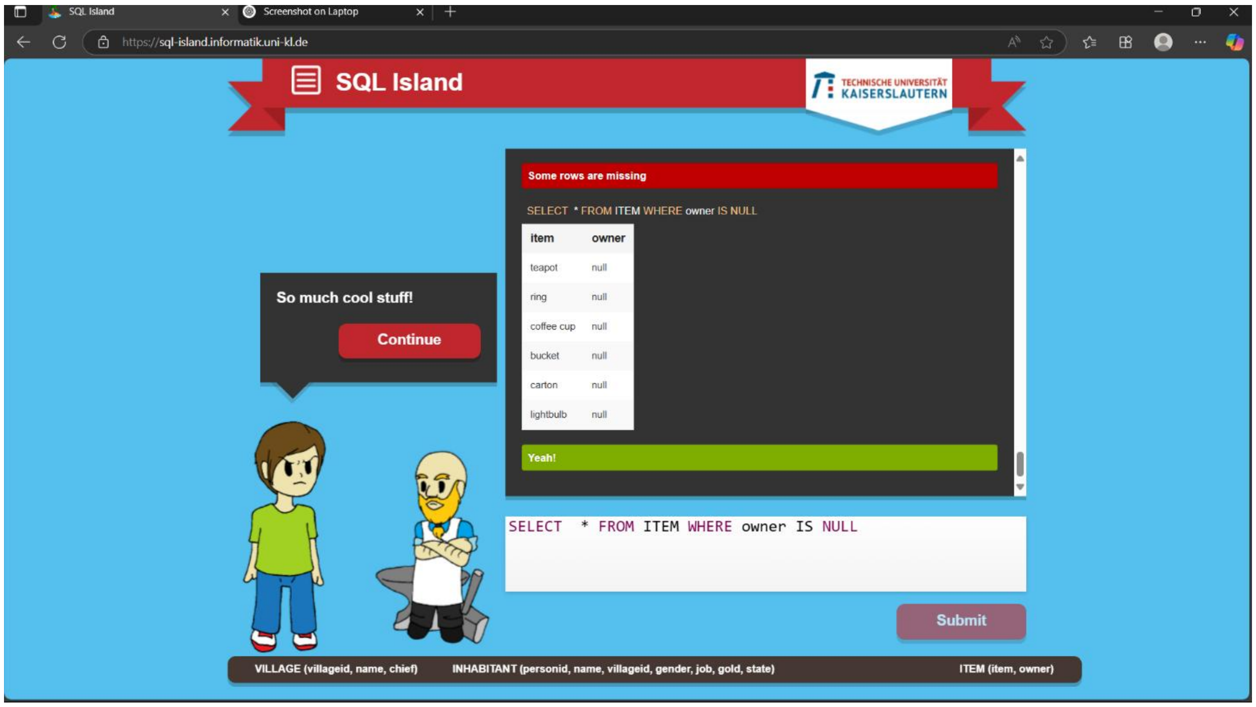
```
SELECT gold FROM INHABITANT WHERE name = "Stranger";
gold
0
```

Yeah!

```
SELECT gold FROM INHABITANT WHERE name = "Stranger";
```

SELECT gold FROM inhabitant WHERE name= 'Stranger';

This screenshot shows the query that retrieves how much gold the inhabitant 'Stranger' has.



Some rows are missing

```
SELECT  * FROM ITEM WHERE owner IS NULL
```

item	owner
teapot	null
ring	null
coffee cup	null
bucket	null
carton	null
lightbulb	null

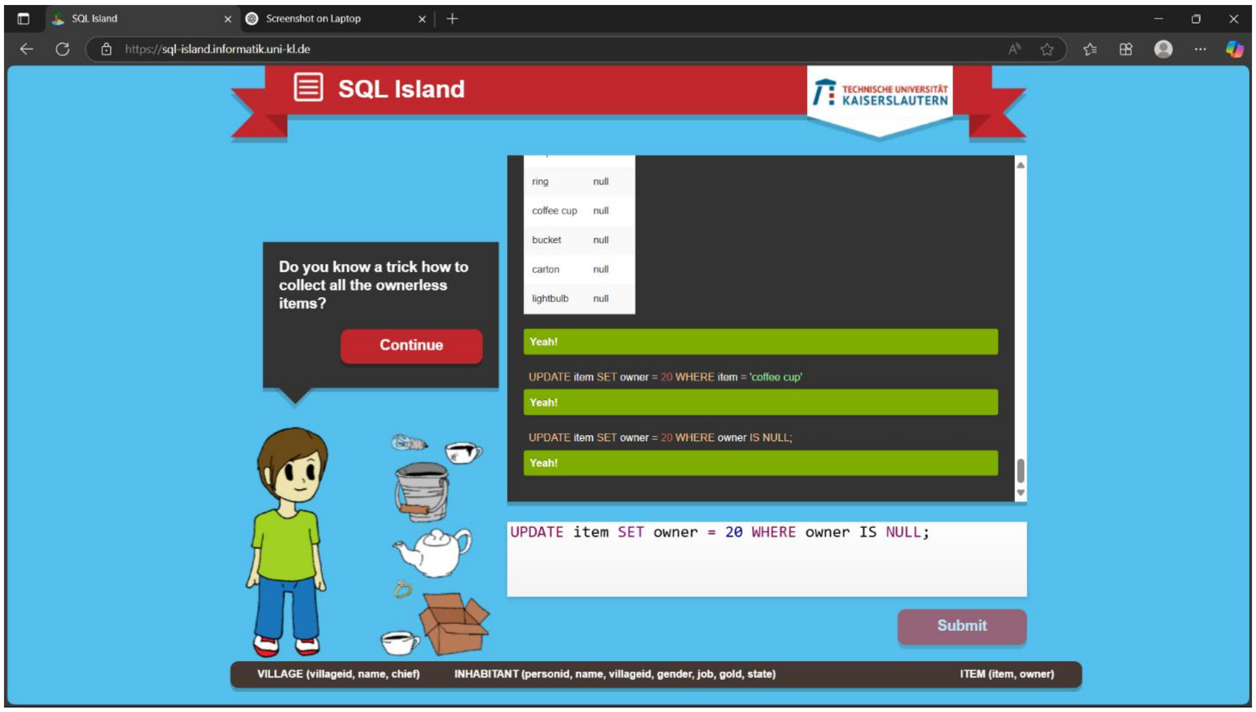
Yeah!

So much cool stuff!

Continue
Submit
SQL Island
TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
https://sql-island.informatik.uni-kl.de
VILLAGE (villageid, name, chief)
INHABITANT (personid, name, villageid, gender, job, gold, state)
ITEM (item, owner)

SELECT * FROM ITEM WHERE owner IS NULL;

This screenshot shows my query that displays all items without an owner.



Do you know a trick how to collect all the ownerless items?

ring	null
coffee cup	null
bucket	null
carton	null
lightbulb	null

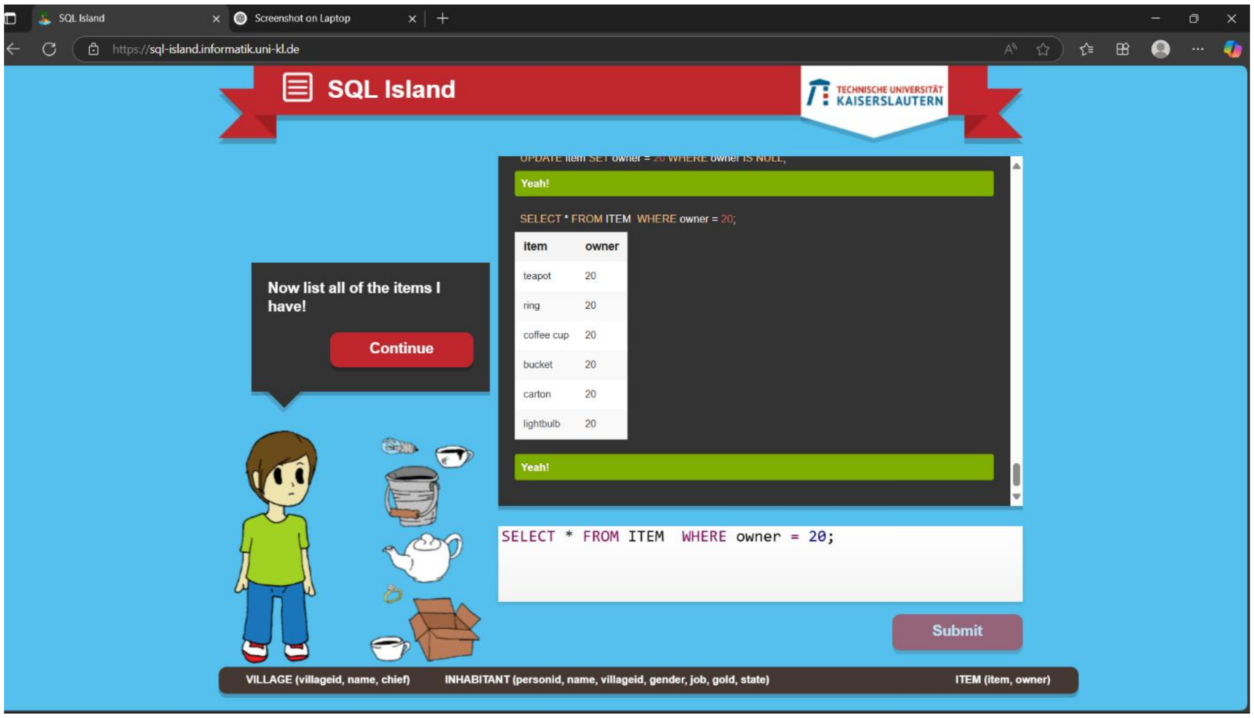
```
Yeah!
UPDATE item SET owner = 20 WHERE item = 'coffee cup'
Yeah!
UPDATE item SET owner = 20 WHERE owner IS NULL;
Yeah!
```

```
UPDATE item SET owner = 20 WHERE owner IS NULL;
```

VILLAGE (villageid, name, chief) INHABITANT (personid, name, villageid, gender, job, gold, state) ITEM (item, owner)

UPDATE item SET owner=20 WHERE owner IS NULL;

This screenshot shows my update query that assigns owner ID 20 to all unowned items.


SQL Island
Screenshot on Laptop
https://sql-island.informatik.uni-kl.de
SQL Island
TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
Now list all of the items I have!
Continue
Submit
VILLAGE (villageid, name, chief)
INHABITANT (personid, name, villageid, gender, job, gold, state)
ITEM (item, owner)

```
UPDATE item SET owner = 20 WHERE owner IS NULL;
Yeah!

SELECT * FROM ITEM  WHERE owner = 20;
```

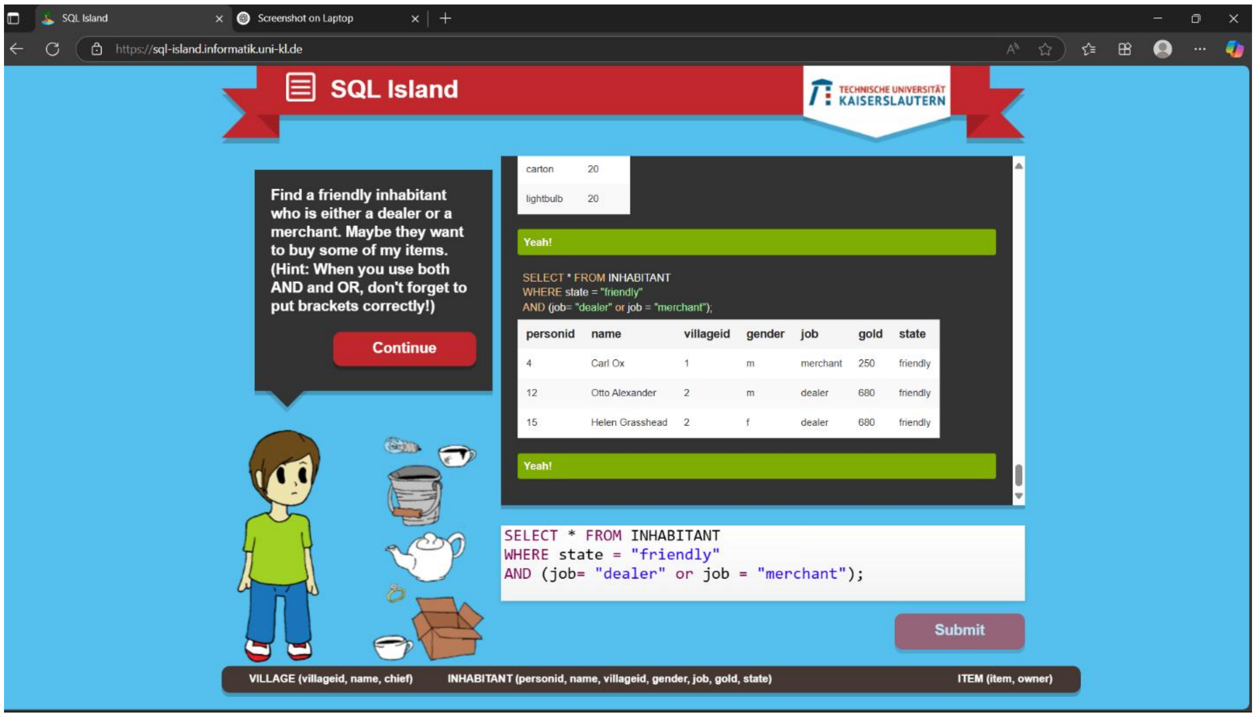
item	owner
teapot	20
ring	20
coffee cup	20
bucket	20
carton	20
lightbulb	20

```
Yeah!
```

```
SELECT * FROM ITEM  WHERE owner = 20;
```

SELECT * FROM ITEM WHERE owner = 20;

This screenshot shows a query verifying that the items have been assigned to owner ID 20.


SQL Island
https://sql-island.informatik.uni-kl.de
TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
Continue
Submit

Find a friendly inhabitant who is either a dealer or a merchant. Maybe they want to buy some of my items. (Hint: When you use both AND and OR, don't forget to put brackets correctly!)

carton	20
lightbulb	20

Yeah!

```
SELECT * FROM INHABITANT
WHERE state = "friendly"
AND (job= "dealer" or job = "merchant");
```

personid	name	villageid	gender	job	gold	state
4	Carl Ox	1	m	merchant	250	friendly
12	Otto Alexander	2	m	dealer	680	friendly
15	Helen Grasshead	2	f	dealer	680	friendly

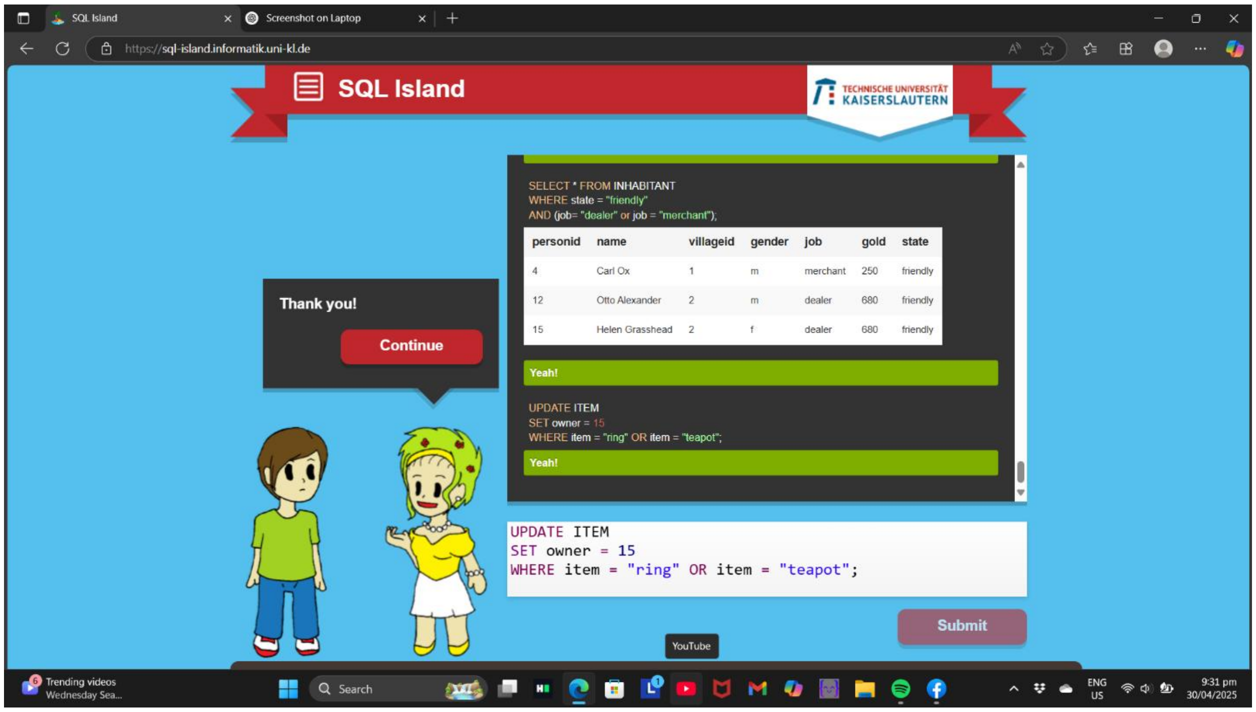
Yeah!

```
SELECT * FROM INHABITANT
WHERE state = "friendly"
AND (job= "dealer" or job = "merchant");
```

VILLAGE (villageid, name, chief)   INHABITANT (personid, name, villageid, gender, job, gold, state)   ITEM (item, owner)

FROM inhabitant WHERE state= 'friendly' AND (job = 'dealer' OR job= 'merchant');

This screenshot shows my query to find friendly inhabitants who are either dealers or merchants.



```
SELECT * FROM INHABITANT
WHERE state = "friendly"
AND (job= "dealer" or job = "merchant");
```

personid	name	villageid	gender	job	gold	state
4	Carl Ox	1	m	merchant	250	friendly
12	Otto Alexander	2	m	dealer	680	friendly
15	Helen Grasshead	2	f	dealer	680	friendly

Yeah!

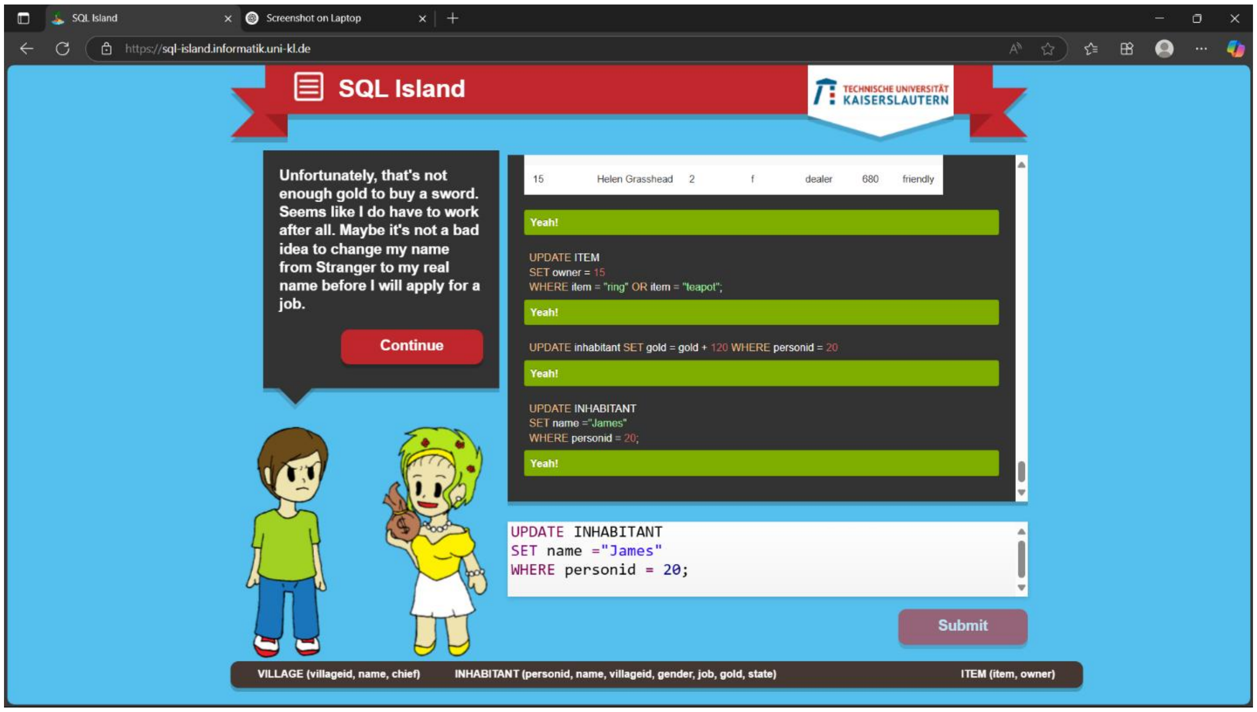
```
UPDATE ITEM
SET owner = 15
WHERE item = "ring" OR item = "teapot";
```

Yeah!

```
UPDATE ITEM
SET owner = 15
WHERE item = "ring" OR item = "teapot";
```

UPDATE item SET owner=15 WHERE item = "ring" OR item = "teapot";

This screenshot shows an update query to assign the 'ring' and 'teapot' items to owner ID 15.


SQL Island
https://sql-island.informatik.uni-kl.de
Technische Universität Kaiserslautern
Continue
Submit

Unfortunately, that's not enough gold to buy a sword. Seems like I do have to work after all. Maybe it's not a bad idea to change my name from Stranger to my real name before I will apply for a job.

```
15    Helen Grasshead    2    f    dealer    680    friendly
```

Yeah!

```
UPDATE ITEM
SET owner = 15
WHERE item = "ring" OR item = "teapot";
```

Yeah!

```
UPDATE inhabitant SET gold = gold + 120 WHERE personid = 20
```

Yeah!

```
UPDATE INHABITANT
SET name ="James"
WHERE personid = 20;
```

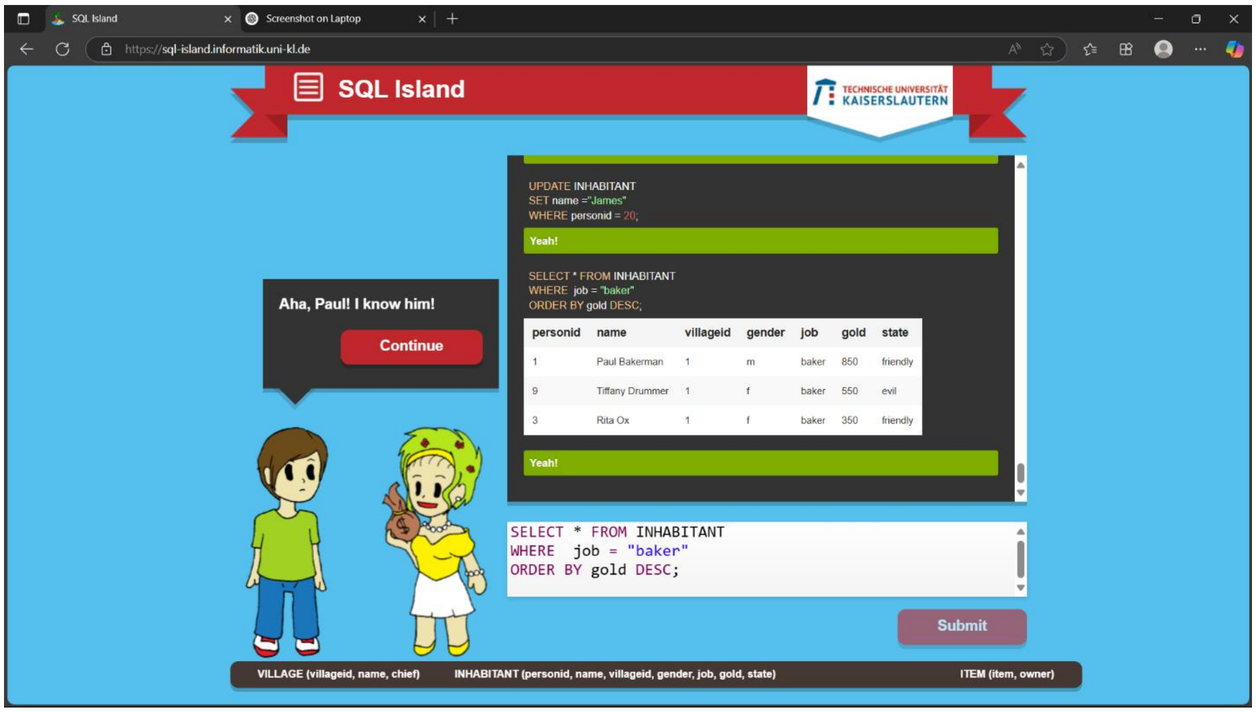
Yeah!

```
UPDATE INHABITANT
SET name ="James"
WHERE personid = 20;
```

VILLAGE (villageid, name, chief) INHABITANT (personid, name, villageid, gender, job, gold, state) ITEM (item, owner)

UPDATE inhabitant SET name= "James" WHERE personid=20;

This screenshot shows my query updating the name of the inhabitant with person ID 20 to "James".


SQL Island
Screenshot on Laptop
https://sql-island.informatik.uni-kl.de
TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
Aha, Paul! I know him!
Continue
Submit

```
UPDATE INHABITANT
SET name ="James"
WHERE personid = 20;
```

Yeah!

```
SELECT * FROM INHABITANT
WHERE  job = "baker"
ORDER BY gold DESC;
```

personid	name	villageid	gender	job	gold	state
1	Paul Bakerman	1	m	baker	850	friendly
9	Tiffany Drummer	1	f	baker	550	evil
3	Rita Ox	1	f	baker	350	friendly

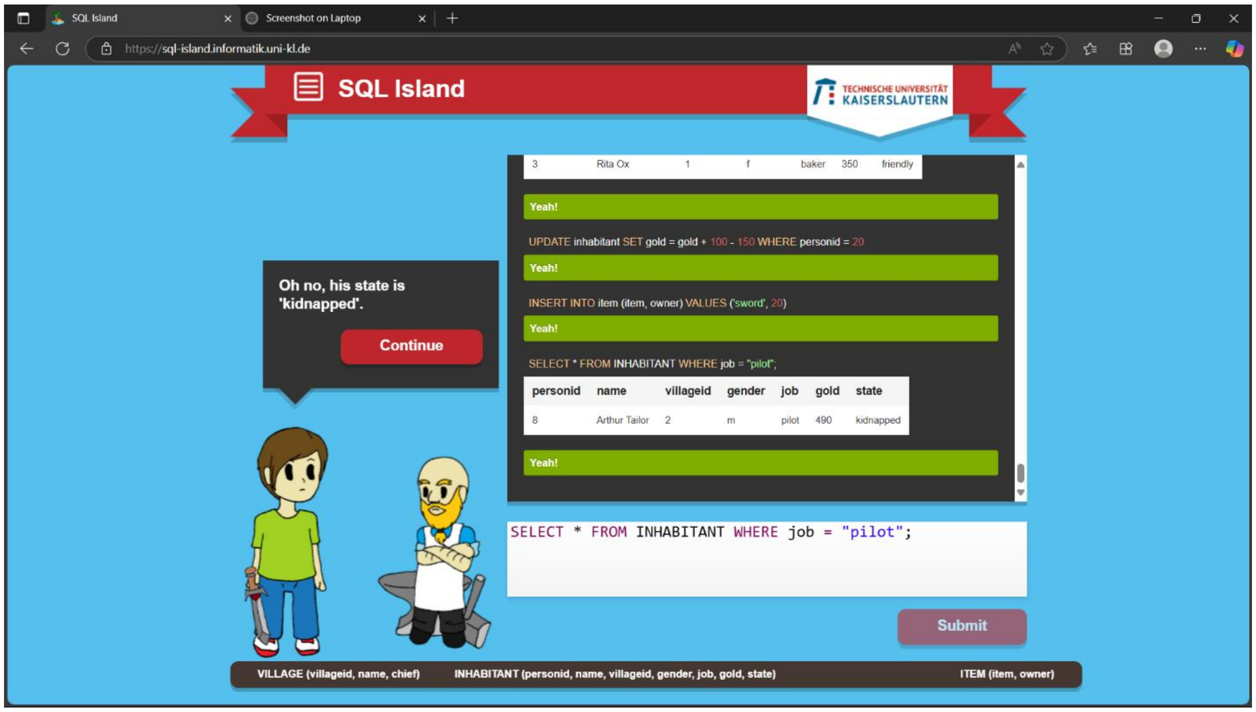
Yeah!

```
SELECT * FROM INHABITANT
WHERE  job = "baker"
ORDER BY gold DESC;
```

VILLAGE (villageid, name, chief) INHABITANT (personid, name, villageid, gender, job, gold, state) ITEM (item, owner)

SELECT * FROM inhabitant WHERE job='baker' ORDER BY gold DESC;

This screenshot shows my query listing bakers ordered by their amount of gold, from highest to lowest.


SQL Island
Screenshot on Laptop
https://sql-island.informatik.uni-kl.de
SQL Island
TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
Continue
Submit
VILLAGE (villageid, name, chief)
INHABITANT (personid, name, villageid, gender, job, gold, state)
ITEM (item, owner)

Oh no, his state is 'kidnapped'.

3	Rita Ox	1	f	baker	350	friendly

Yeah!

```
UPDATE inhabitant SET gold = gold + 100 - 150 WHERE personid = 20
```

Yeah!

```
INSERT INTO item (item, owner) VALUES ('sword', 20)
```

Yeah!

```
SELECT * FROM INHABITANT WHERE job = "pilot";
```

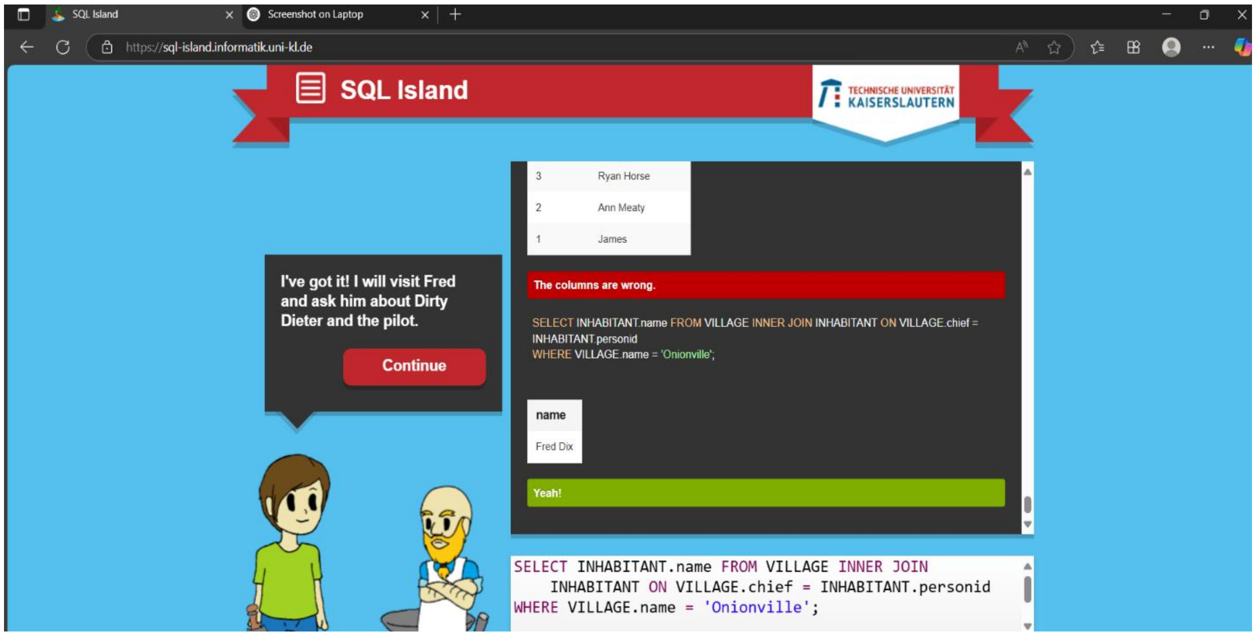
personid	name	villageid	gender	job	gold	state
8	Arthur Tailor	2	m	pilot	490	kidnapped

Yeah!

```
SELECT * FROM INHABITANT WHERE job = "pilot";
```

SELECT * FROM inhabitant WHERE job='pilot';

This screenshot shows my query to find all inhabitants with the job 'pilot'.


SQL Island
TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
I've got it! I will visit Fred and ask him about Dirty Dieter and the pilot.
Continue

3	Ryan Horse
2	Ann Meaty
1	James

The columns are wrong.

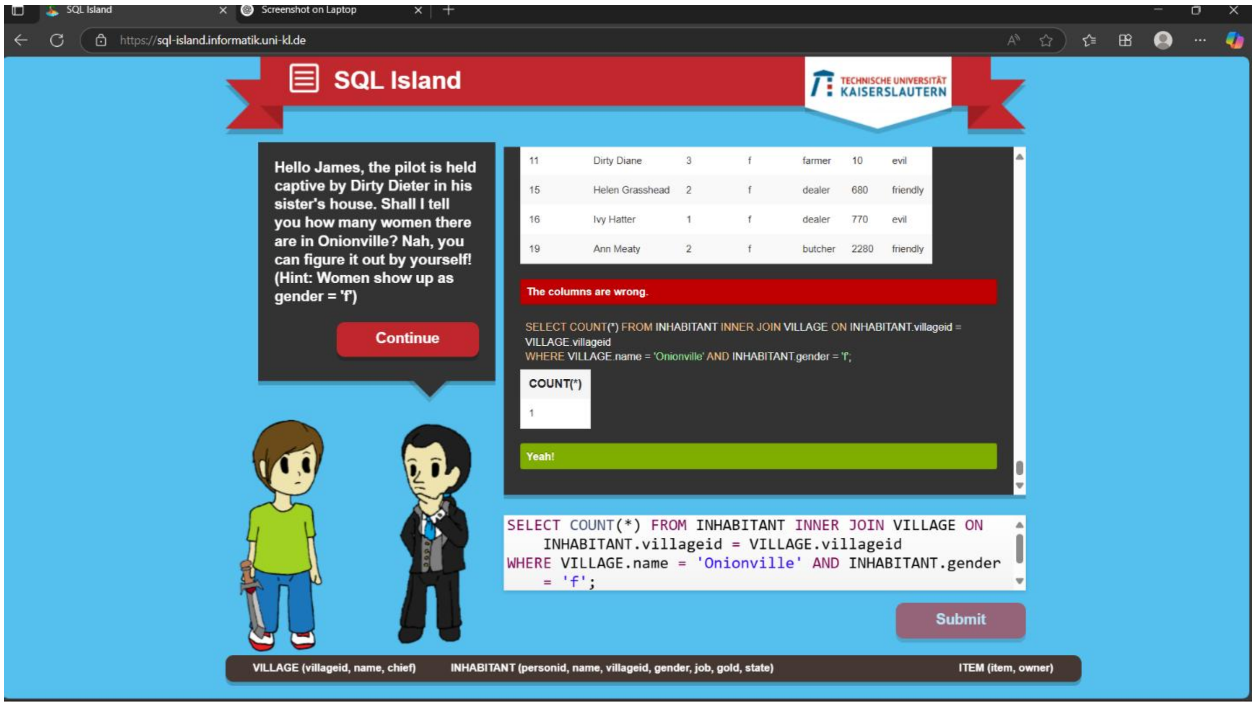
```
SELECT INHABITANT.name FROM VILLAGE INNER JOIN INHABITANT ON VILLAGE.chief =
INHABITANT.personid
WHERE VILLAGE.name = 'Onionville';
```

name
Fred Dix

Yeah!

```
SELECT INHABITANT.name FROM VILLAGE INNER JOIN
    INHABITANT ON VILLAGE.chief = INHABITANT.personid
WHERE VILLAGE.name = 'Onionville';
```

SELECT INHABITANT.name FROM VILLAGE INNER JOIN INHABITANT ON VILLAGE.chief = INHABITANT.personid WHERE VILLAGE.name = 'Onionville';. This will get the **name of the chief** of the village **Onionville**



Hello James, the pilot is held captive by Dirty Dieter in his sister's house. Shall I tell you how many women there are in Onionville? Nah, you can figure it out by yourself! (Hint: Women show up as gender = 'f')

11	Dirty Diane	3	f	farmer	10	evil
15	Helen Grasshead	2	f	dealer	680	friendly
16	Ivy Hatter	1	f	dealer	770	evil
19	Ann Meaty	2	f	butcher	2280	friendly

The columns are wrong.

```
SELECT COUNT(*) FROM INHABITANT INNER JOIN VILLAGE ON INHABITANT.villageid =
VILLAGE.villageid
WHERE VILLAGE.name = 'Onionville' AND INHABITANT.gender = 'f';
```

COUNT(*)
1

Yeah!

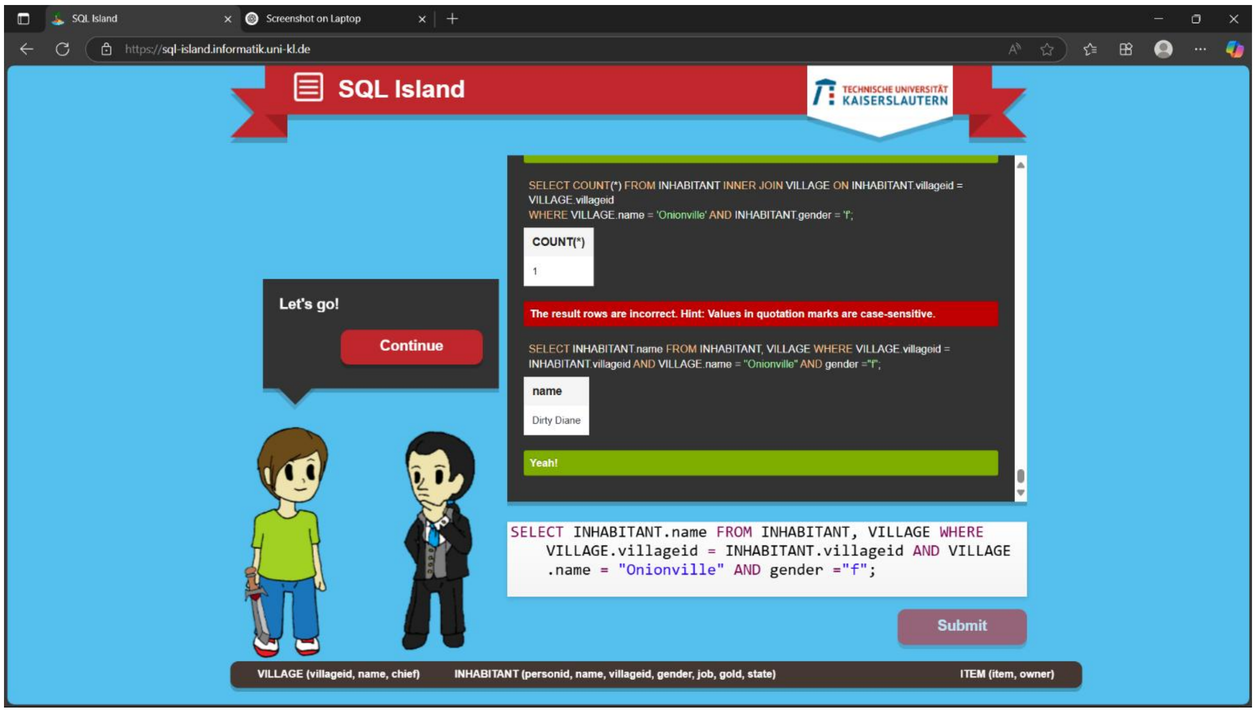
```
SELECT COUNT(*) FROM INHABITANT INNER JOIN VILLAGE ON
    INHABITANT.villageid = VILLAGE.villageid
WHERE VILLAGE.name = 'Onionville' AND INHABITANT.gender
    = 'f';
```

VILLAGE (villageid, name, chief)   INHABITANT (personid, name, villageid, gender, job, gold, state)   ITEM (item, owner)

SELECT COUNT (*) FROM INHABITANT INNER JOIN VILLAGE ON INHABITANT.villageid = VILLAGE.villageid

VILLAGE.name = 'Onionville' AND INHABITANT.gender = 'f';

This screenshot shows my query to count how many female inhabitants are in Onionville.


SQL Island
Screenshot on Laptop
https://sql-island.informatik.uni-kl.de
SQL Island
TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
Let's go!
Continue
Submit
VILLAGE (villageid, name, chief)
INHABITANT (personid, name, villageid, gender, job, gold, state)
ITEM (item, owner)

```
SELECT COUNT(*) FROM INHABITANT INNER JOIN VILLAGE ON INHABITANT.villageid =
VILLAGE.villageid
WHERE VILLAGE.name = 'Onionville' AND INHABITANT.gender = 'f';
```

COUNT(*)
1

The result rows are incorrect. Hint: Values in quotation marks are case-sensitive.

```
SELECT INHABITANT.name FROM INHABITANT, VILLAGE WHERE VILLAGE.villageid =
INHABITANT.villageid AND VILLAGE.name = "Onionville" AND gender ="f";
```

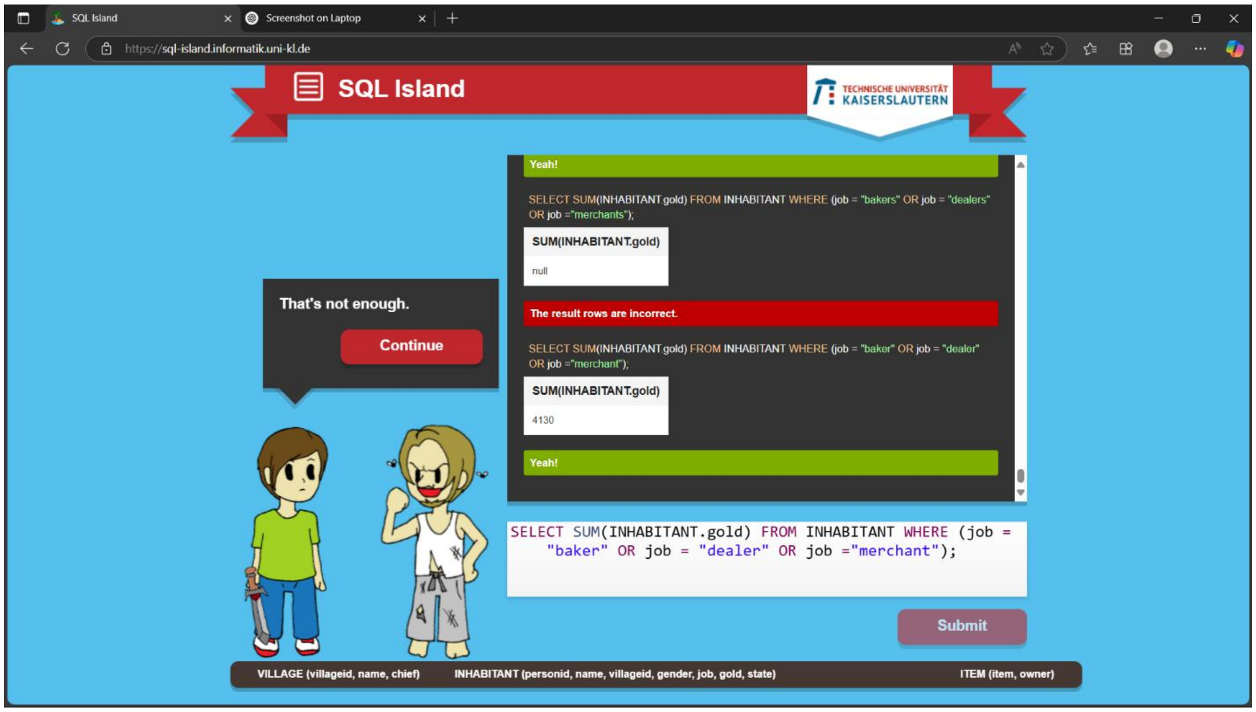
name
Dirty Diane

Yeah!

```
SELECT INHABITANT.name FROM INHABITANT, VILLAGE WHERE
    VILLAGE.villageid = INHABITANT.villageid AND VILLAGE
    .name = "Onionville" AND gender ="f";
```

SELECT INHABITANT.name FROM INHABITANT, VILLAGE WHERE VILLAGE.villageid = INHABITANT.villageid AND VILLAGE.name = “ Onionville” AND gender=”f”;

This screenshot shows my query listing the names of female inhabitants in Onionville.


SQL Island
Technische Universität Kaiserslautern
That's not enough.
Continue
Submit
VILLAGE (villageid, name, chief)
INHABITANT (personid, name, villageid, gender, job, gold, state)
ITEM (item, owner)

Yeah!

```
SELECT SUM(INHABITANT.gold) FROM INHABITANT WHERE (job = "bakers" OR job = "dealers"
OR job ="merchants");
```

SUM(INHABITANT.gold)
null

The result rows are incorrect.

```
SELECT SUM(INHABITANT.gold) FROM INHABITANT WHERE (job = "baker" OR job = "dealer"
OR job ="merchant");
```

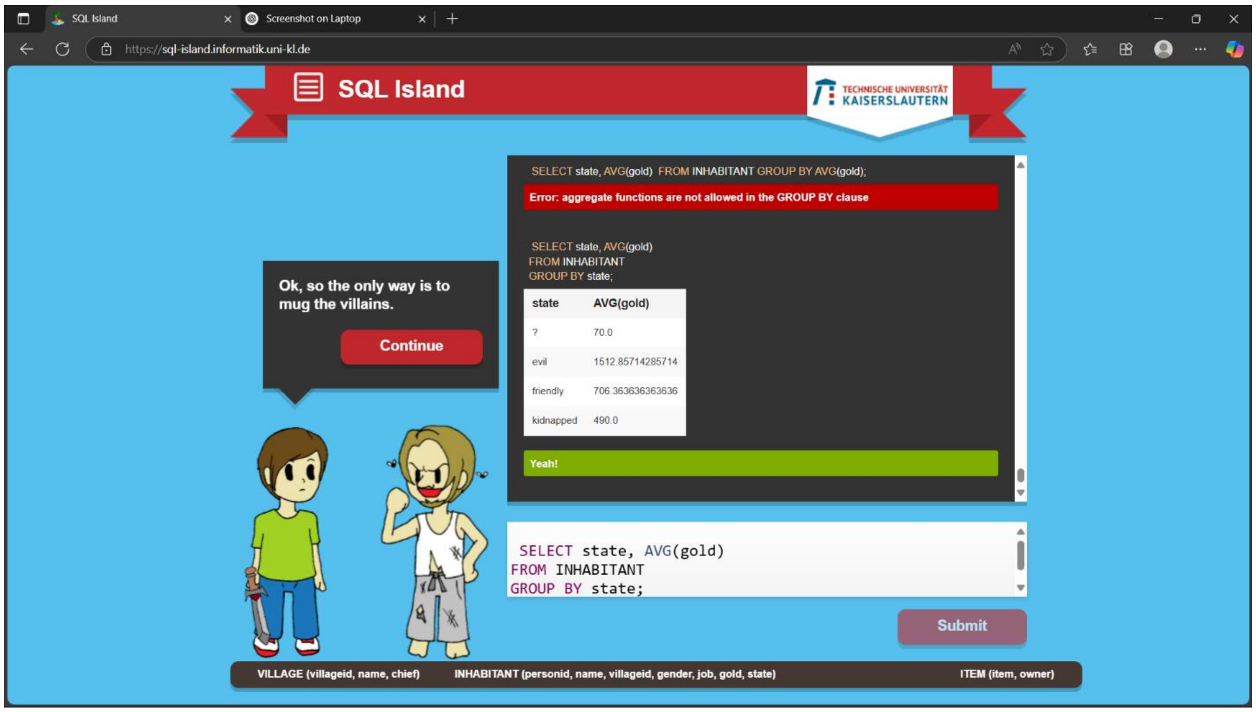
SUM(INHABITANT.gold)
4130

Yeah!

```
SELECT SUM(INHABITANT.gold) FROM INHABITANT WHERE (job =
    "baker" OR job = "dealer" OR job ="merchant");
```

SELECT SUM(inhabitant.gold) FROM inhabitant WHERE (job='baker' OR job= 'dealer' OR job='merchant');

This screenshot shows my query calculating the total gold owned by all bakers, dealers, and merchants.



```
SELECT state, AVG(gold)  FROM INHABITANT GROUP BY AVG(gold);
```

```
Error: aggregate functions are not allowed in the GROUP BY clause
```

```
SELECT state, AVG(gold)
FROM INHABITANT
GROUP BY state;
```

state	AVG(gold)
?	70.0
evil	1512.85714285714
friendly	706.363636363636
kidnapped	490.0

```
Yeah!
```

Ok, so the only way is to mug the villains.

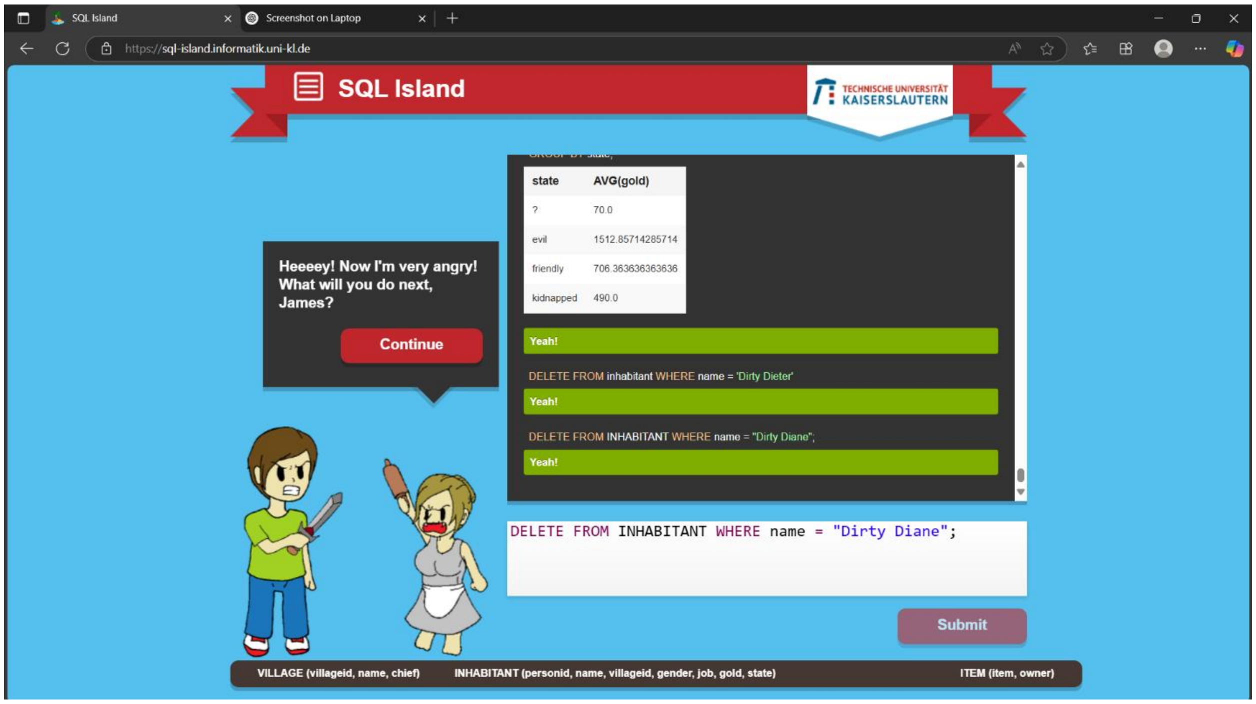
```
SELECT state, AVG(gold)
FROM INHABITANT
GROUP BY state;
```

VILLAGE (villageid, name, chief) INHABITANT (personid, name, villageid, gender, job, gold, state) ITEM (item, owner)

SQL Island
https://sql-island.informatik.uni-kl.de
TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
Continue
Submit

SELECT state, AVG (gold) FROM inhabitant GROUP BY state ORDER BY AVG (gold);

This screenshot shows my query to get the average gold amount for each state, ordered by average gold.


SQL Island
https://sql-island.informatik.uni-kl.de
TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
Continue
Submit

state	AVG(gold)
?	70.0
evil	1512.85714285714
friendly	706.363636363636
kidnapped	490.0

Heeeey! Now I'm very angry! What will you do next, James?

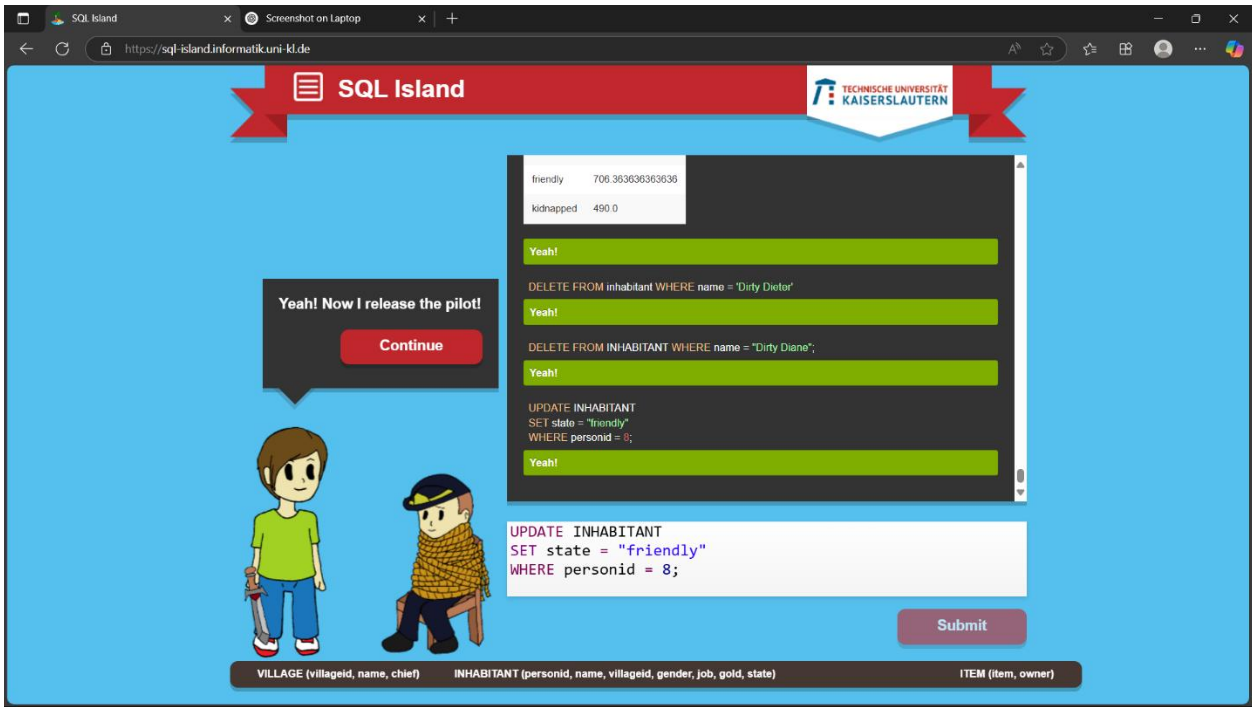
```
Yeah!
DELETE FROM inhabitant WHERE name = 'Dirty Dieter'
Yeah!
DELETE FROM INHABITANT WHERE name = "Dirty Diane";
Yeah!
```

```
DELETE FROM INHABITANT WHERE name = "Dirty Diane";
```

VILLAGE (villageid, name, chief) INHABITANT (personid, name, villageid, gender, job, gold, state) ITEM (item, owner)

DELETE FROM inhabitant WHERE name = 'Dirty Diane';

This screenshot shows my query deleting the inhabitant named 'Dirty Diane'.


SQL Island
Screenshot on Laptop
https://sql-island.informatik.uni-kl.de
SQL Island
TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
Yeah! Now I release the pilot!
Continue
Submit

friendly	706.363636363636
kidnapped	490.0

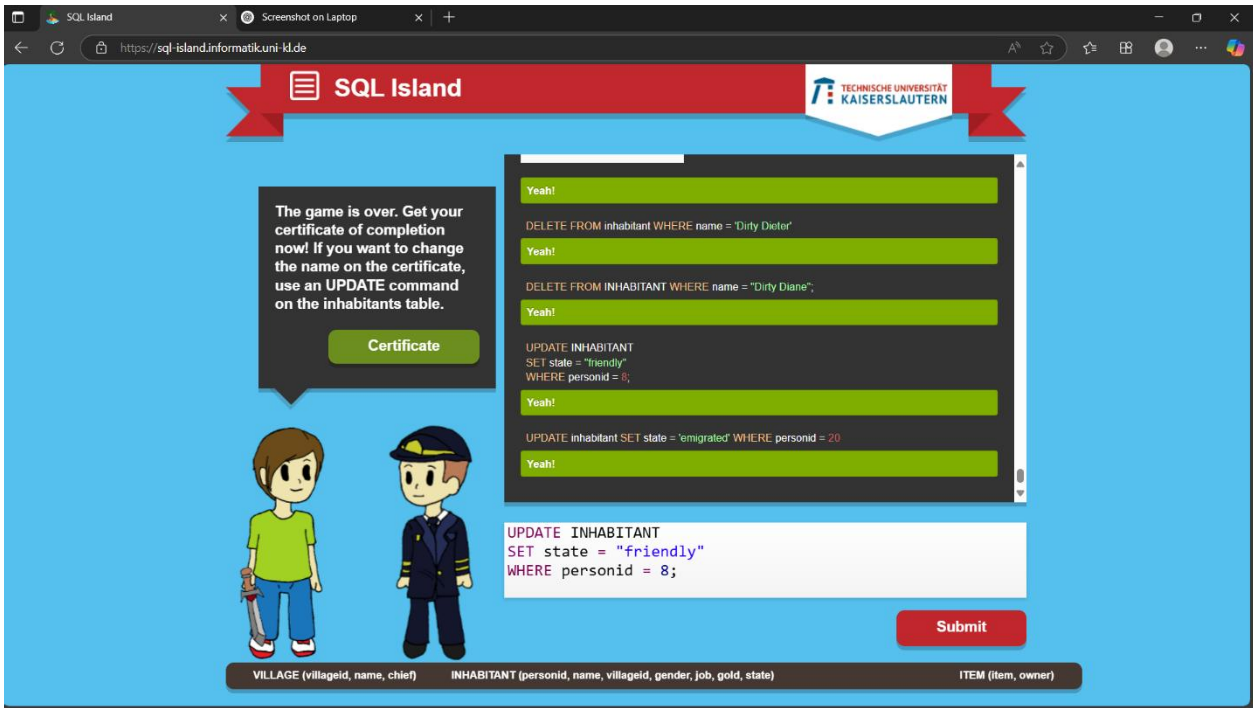
```
Yeah!
DELETE FROM inhabitant WHERE name = 'Dirty Dieter'
Yeah!
DELETE FROM INHABITANT WHERE name = "Dirty Diane";
Yeah!
UPDATE INHABITANT
SET state = "friendly"
WHERE personid = 8;
Yeah!
```

```
UPDATE INHABITANT
SET state = "friendly"
WHERE personid = 8;
```

VILLAGE (villageid, name, chief)   INHABITANT (personid, name, villageid, gender, job, gold, state)   ITEM (item, owner)

UPDATE inhabitant SET state ='friendly' WHERE personid=8;

This screenshot shows my query to update the state of the inhabitant with person ID 8 to 'friendly'.


SQL Island
Screenshot on Laptop
https://sql-island.informatik.uni-kl.de
SQL Island
TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
Certificate
Submit

The game is over. Get your certificate of completion now! If you want to change the name on the certificate, use an UPDATE command on the inhabitants table.

Yeah!

```
DELETE FROM inhabitant WHERE name = 'Dirty Dieter'
```

Yeah!

```
DELETE FROM INHABITANT WHERE name = "Dirty Diane";
```

Yeah!

```
UPDATE INHABITANT
SET state = "friendly"
WHERE personid = 8;
```

Yeah!

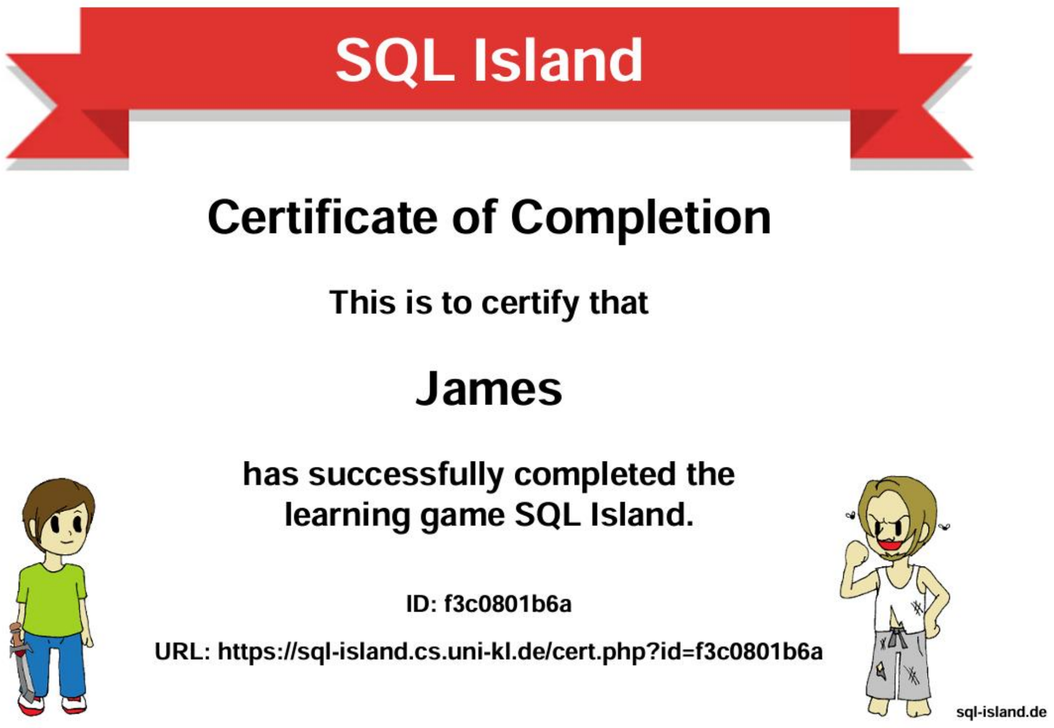
```
UPDATE inhabitant SET state = 'emigrated' WHERE personid = 20
```

Yeah!

```
UPDATE INHABITANT
SET state = "friendly"
WHERE personid = 8;
```

VILLAGE (villageid, name, chief) INHABITANT (personid, name, villageid, gender, job, gold, state) ITEM (item, owner)

" COMPLETION CERTIFICATE FOR SQL ISLAND"



SQL Island

Certificate of Completion

This is to certify that

James

has successfully completed the
learning game SQL Island.

ID: f3c0801b6a

URL: https://sql-island.cs.uni-kl.de/cert.php?id=f3c0801b6a

sql-island.de