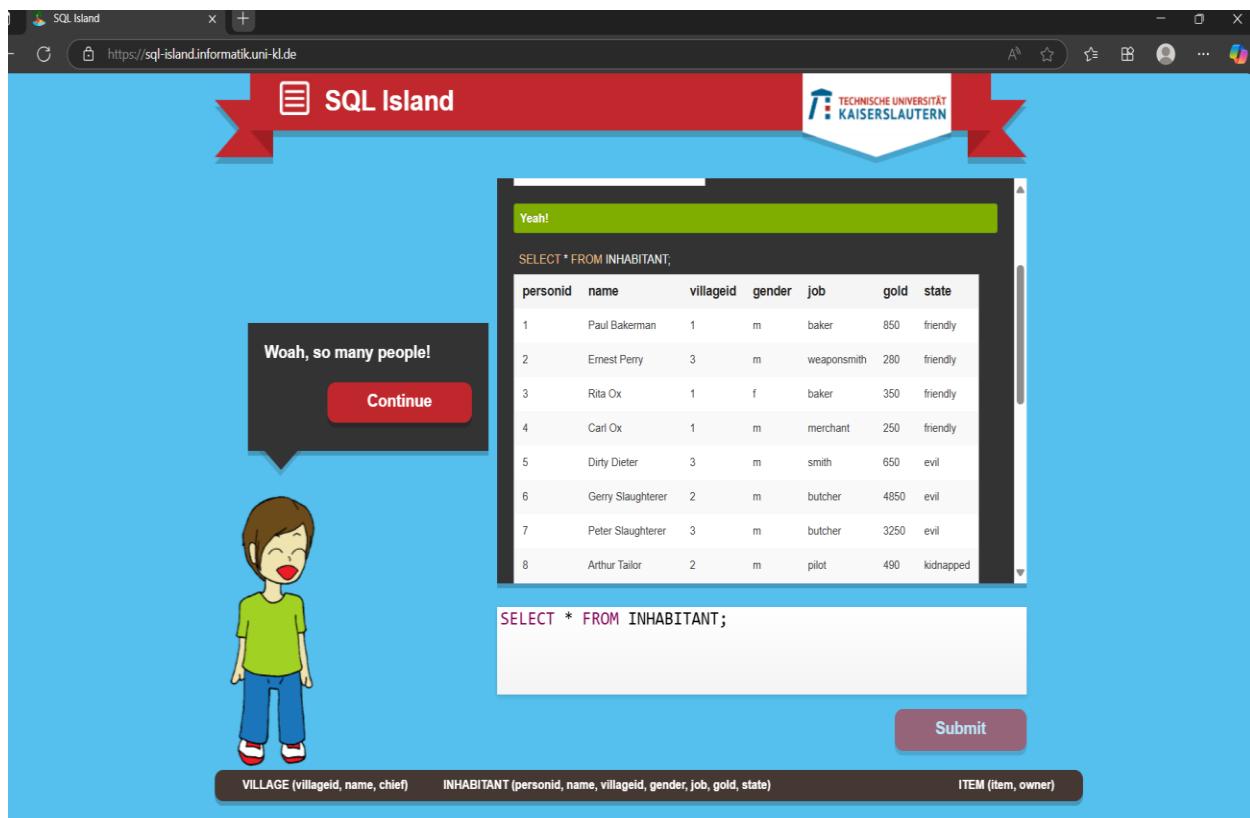


SQL ISLAND SEMI-PROJECT

Name: James Olandria Cuso

Date: May 2, 2025



SELECT * FROM INHABITANT;

This will display all the inhabitants from the INHABITANT table.

The screenshot shows a web-based game interface titled "SQL Island". At the top, there's a red banner with the title "SQL Island" and the logo of Technische Universität Kaiserslautern. Below the banner, a character named Edward is shown with a speech bubble saying "Thank you, Edward! Okay, let's see who is friendly on this island...". A "Continue" button is below the speech bubble. To the right, there's a table titled "INHABITANT" with columns: personid, name, villageid, gender, job, gold, and state. The table contains 10 rows of data. Below the table is a code input field containing the SQL query "SELECT * FROM INHABITANT WHERE state = 'friendly';" and a "Submit" button. At the bottom, there are three buttons: "VILLAGE (villageid, name, chief)", "INHABITANT (personid, name, villageid, gender, job, gold, state)", and "ITEM (item, owner)".

personid	name	villageid	gender	job	gold	state
1	Paul Bakeman	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 Mooty	2	f	butcher	2380	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.

The screenshot shows a browser window for 'SQL Island' at <https://sql-island.informatik.uni-kl.de>. The page features a red banner with the title 'SQL Island' and the Technische Universität Kaiserslautern logo. A speech bubble contains a hint: 'There is no way around getting a sword for myself. I will now try to find a friendly weapon smith to forge me one. (Hint: You can combine predicates in the WHERE clause with AND)'. Below it is a 'Continue' button. To the right, there are two tables: 'INHABITANT' and 'ITEM'. The 'INHABITANT' table shows data for five characters:

	personid	name	villageid	gender	job	gold	state
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	

A green bar says 'Yeah!'. The 'ITEM' table shows data for one item:

item	owner
2	Ernest Perry

A green bar says 'Yeah!'. Below the tables is a SQL query input field containing:

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

At the bottom are three buttons: 'VILLAGE (villageid, name, chief)', 'INHABITANT (personid, name, villageid, gender, job, gold, state)', and 'ITEM (item, owner)'. There is also a 'Submit' button.

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

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

The screenshot shows a web-based game interface for learning SQL. At the top, there's a red banner with the text "SQL Island" and the Technische Universität Kaiserslautern logo. Below the banner, a character on the left says, "That looks better! I will go and visit those smiths." with a "Continue" button. On the right, a modal window displays a query result:

```
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

Below the table, another modal window shows a query:

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

At the bottom, there are three buttons: "VILLAGE (villageid, name, chief)", "INHABITANT (personid, name, villageid, gender, job, gold, state)", and "ITEM (item, owner)". A "Submit" button is also present.

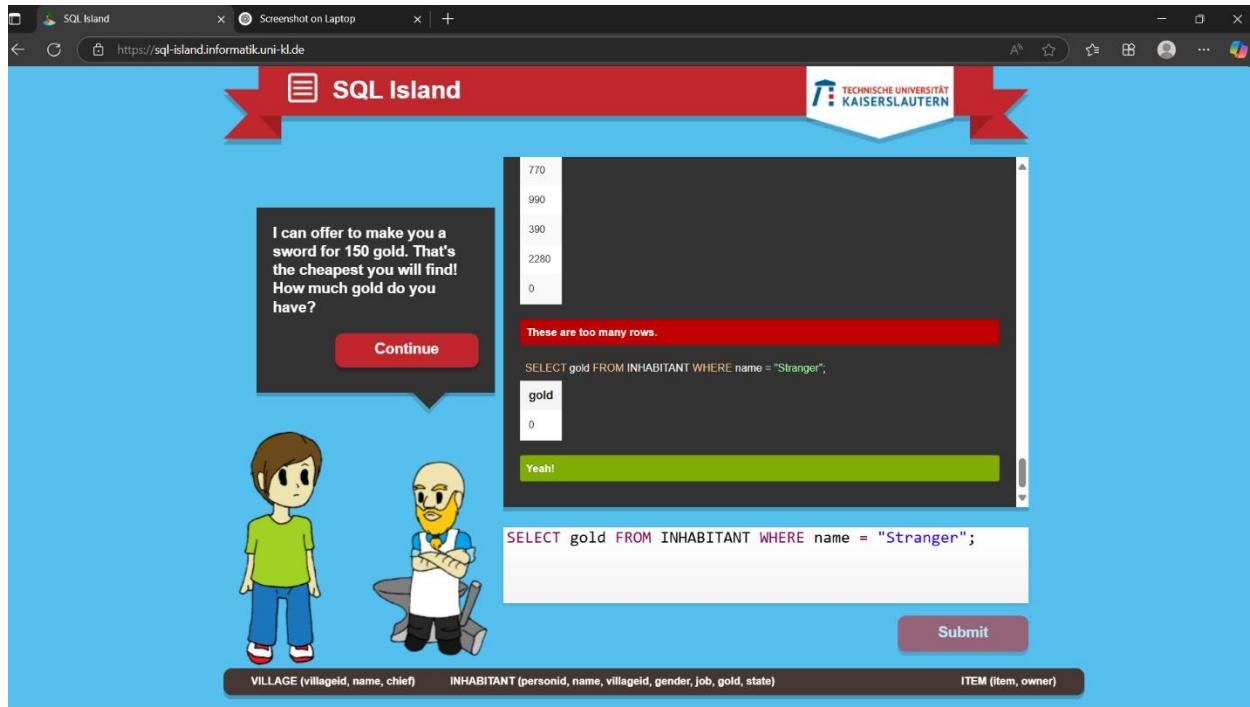
```
SELECT * FROM INHABITANT WHERE state='friendly' AND job LIKE  
'%smith';
```

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

The screenshot shows a web browser window for 'SQL Island' at the URL <https://sql-island.informatik.uni-kl.de>. The page features a red header with the 'SQL Island' logo and the Technische Universität Kaiserslautern logo. A sidebar on the left contains a hint about using a SELECT query without an asterisk to get specific columns. The main area shows a terminal window displaying the results of a query. The output shows multiple rows of 'personid' values (17, 18, 19, 20) and a message 'These are too many rows.' Below the terminal is a text input field containing the query: 'SELECT personid FROM INHABITANT WHERE name = "Stranger";'. A 'Submit' button is located below the input field. At the bottom of the screen, there are three tabs: 'VILLAGE (villageid, name, chief)', 'INHABITANT (personid, name, villageid, gender, job, gold, state)', and 'ITEM (item, owner)'. On the left side, there are two cartoon characters: a boy in a green shirt and blue pants, and an older man in a white shirt and black top hat.

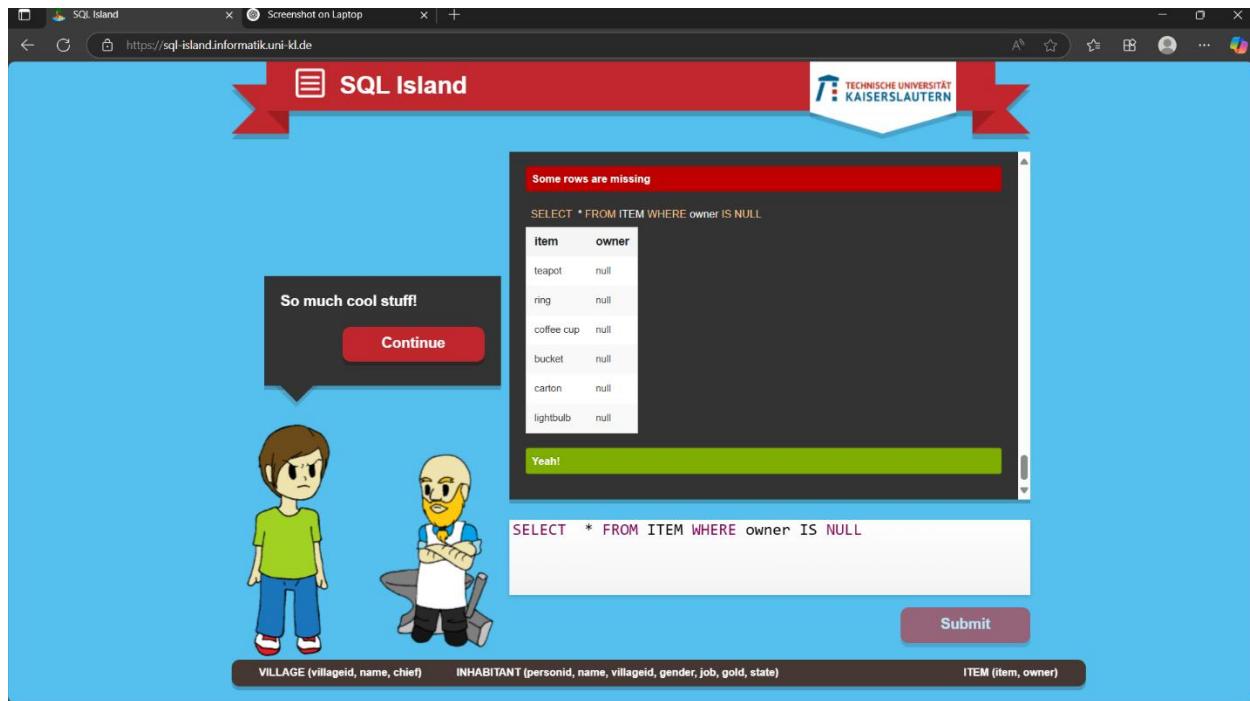
```
SELECT personid FROM INHABITANT WHERE name= 'Stranger';
```

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



```
SELECT gold FROM inhabitant WHERE name= 'Stranger';
```

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



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

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

The screenshot shows a browser window for the SQL Island game. The title bar reads "SQL Island" and "Screenshot on Laptop". The URL is "https://sql-island.informatik.uni-kl.de". The main interface features a red banner at the top with the "SQL Island" logo. Below it is a blue background with a cartoon character of a boy in a green shirt and blue pants standing next to several floating items: a ring, a coffee cup, a bucket, a carton, and a lightbulb. A speech bubble from the character asks, "Do you know a trick how to collect all the ownerless items?" with a "Continue" button. To the right is a code editor window with a sidebar showing the current state of the "ITEM" table:

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

The code editor contains three lines of SQL code:

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

Below the code editor is a text input field containing the query:

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

A "Submit" button is located at the bottom right of the code editor.

At the bottom of the screen, there are three database schema labels: "VILLAGE (villageid, name, chief)", "INHABITANT (personid, name, villageid, gender, job, gold, state)", and "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.

The screenshot shows a web browser window for 'SQL Island' at the URL <https://sql-island.informatik.uni-kl.de>. The page features a cartoon character of a boy in a green shirt and blue pants standing next to various floating items like a teapot, a ring, a coffee cup, a bucket, a carton, and a lightbulb. A speech bubble from the character says, 'Now list all of the items I have!' with a 'Continue' button. On the right, there's a terminal window showing a MySQL query and its results:

```
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!
```

Below the terminal is another input field with the query 'SELECT * FROM ITEM WHERE owner = 20;' and a 'Submit' button. At the bottom, there are three database schema descriptions: 'VILLAGE (villageid, name, chief)', 'INHABITANT (personid, name, villageid, gender, job, gold, state)', and 'ITEM (item, owner)'. The top right corner of the page displays the logo of Technische Universität Kaiserslautern.

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

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

The screenshot shows a web-based game titled "SQL Island". At the top, there's a red banner with the title "SQL Island" and the Technische Universität Kaiserslautern logo. Below the banner, a character with brown hair and a green shirt is standing next to a collection of items: a carton, a lightbulb, a teapot, a cup, and a box. A speech bubble contains a hint: "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!)" There's a "Continue" button below the hint. To the right, there's a table showing item counts and a query result. The table has two rows:

carton	20
lightbulb	20

A green bar at the bottom says "Yeah!". Below the table is a query:

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

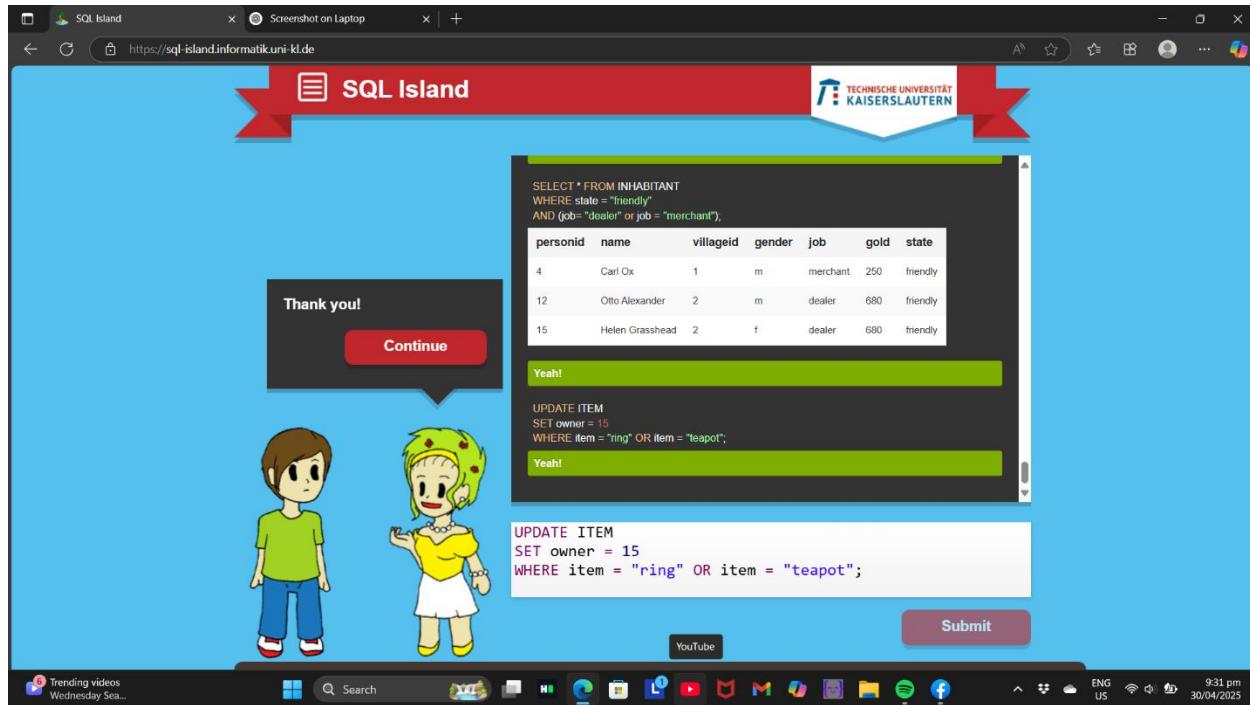
The result of the query is a table:

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

Another green bar at the bottom says "Yeah!". At the bottom right is a "Submit" button. Below the main area are three tabs: "VILLAGE (villageid, name, chief)", "INHABITANT (personid, name, villageid, gender, job, gold, state)", and "ITEM (item, owner)".

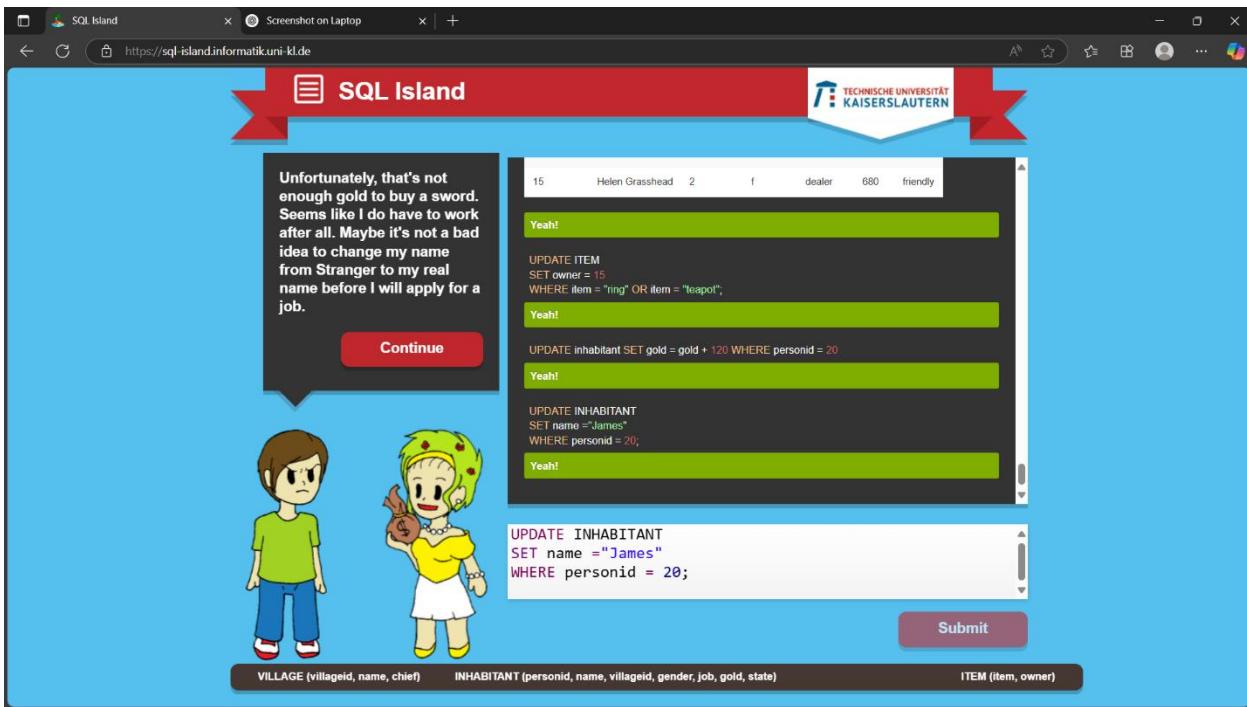
FROM inhabitant WHERE state= 'friendly' AND (job = 'dealer' OR job= 'merchant');

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



`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.



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".

The screenshot shows a browser window for the SQL Island game. The title bar reads "SQL Island" and the URL is "https://sql-island.informatik.uni-kl.de". The game interface has a blue background with a red banner at the top containing the title "SQL Island" and the Technische Universität Kaiserslautern logo.

A character on the left says "Aha, Paul! I know him!" with a "Continue" button below it. In the center, there are two cartoon characters: a boy in a green shirt and blue pants, and a girl with green hair and a yellow dress holding a bun.

The main area contains three code snippets and a table:

- Top snippet:

```
UPDATE INHABITANT  
SET name = "James"  
WHERE personid = 20;
```
- Middle snippet:

```
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
- Bottom snippet:

```
Yeah!
```



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

At the bottom, there are buttons for "VILLAGE (villageid, name, chief)", "INHABITANT (personid, name, villageid, gender, job, gold, state)", "ITEM (item, owner)", and a "Submit" button.

`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.

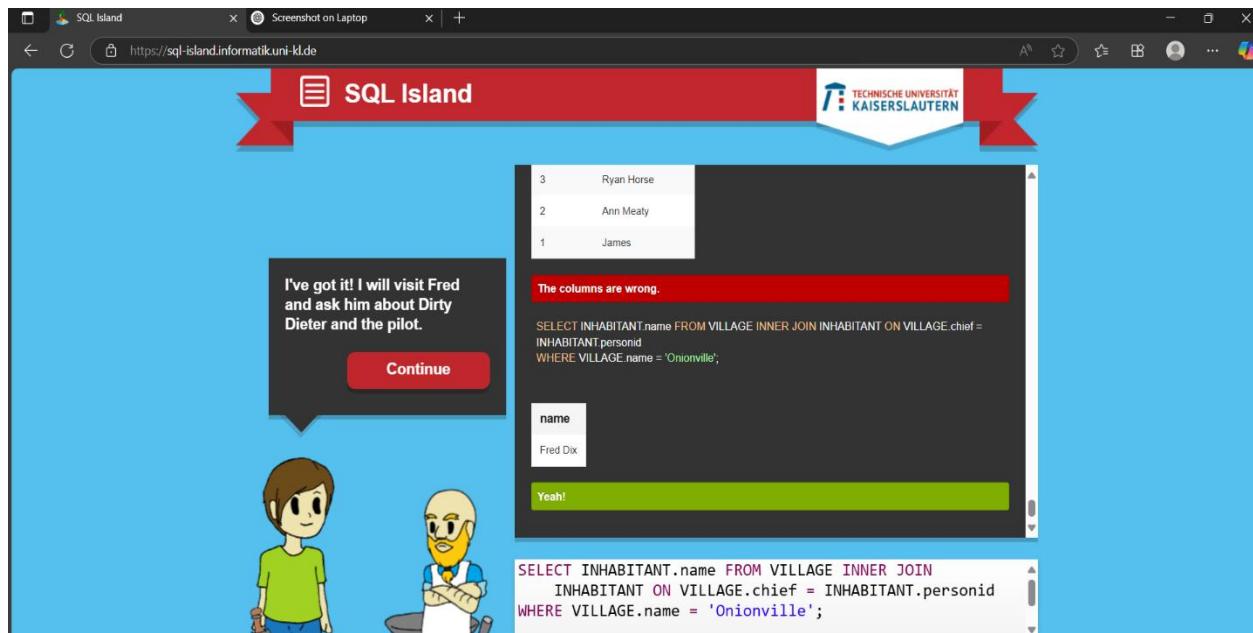
The screenshot shows a browser window for 'SQL Island' at <https://sql-island.informatik.uni-kl.de>. The interface features a red header with the 'SQL Island' logo and the Technische Universität Kaiserslautern logo. On the left, there's a cartoon illustration of two characters: one holding a sword and another with arms crossed. A speech bubble says, 'Oh no, his state is 'kidnapped''. A 'Continue' button is below it. The main area contains a terminal-like window with the following text:

```
3 Rita Ox 1 f baker 360 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";
```

A 'Submit' button is at the bottom right of the terminal window. At the very bottom, there are three tabs: 'VILLAGE (villageid, name, chief)', 'INHABITANT (personid, name, villageid, gender, job, gold, state)', and 'ITEM (item, owner)'.

`SELECT * FROM inhabitant WHERE job='pilot';`

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



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

The screenshot shows a browser window for 'SQL Island' at <https://sql-island.informatik.uni-kl.de>. The page features a red header with the 'SQL Island' logo and the Technische Universität Kaiserslautern logo. A character on the left says: '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')'. A 'Continue' button is below the text. On the right, a table shows data from the INHABITANT and VILLAGE tables:

	name	villageid	gender	job	gold	state
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

A red bar at the bottom of the table area says 'The columns are wrong.' Below the table, two SQL queries are shown:

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

```
COUNT(*)  
1
```

A green bar says 'Yeah!' Below the first query, another query is shown:

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

A 'Submit' button is at the bottom right. At the very bottom, tabs for 'VILLAGE (villageid, name, chief)', 'INHABITANT (personid, name, villageid, gender, job, gold, state)', and 'ITEM (item, owner)' are visible.

```
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.

The screenshot shows a web-based game interface for SQL. At the top, there's a red banner with the text "SQL Island". To the right of the banner is the logo of Technische Universität Kaiserslautern. Below the banner, there's a cartoon illustration of two characters: a boy in a green shirt and blue pants holding a sword, and a girl in a black dress and white blouse holding a book. A speech bubble from the boy says "Let's go!" and a button labeled "Continue" is below it. On the right side, there's a query editor window. It contains three code snippets and their results:

- Query 1:

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

 Result: `COUNT(*)
1` Hint: `The result rows are incorrect. Hint: Values in quotation marks are case-sensitive.`
- Query 2:

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

 Result: `name
Dirty Diane` Hint: `Yeah!`
- Query 3:

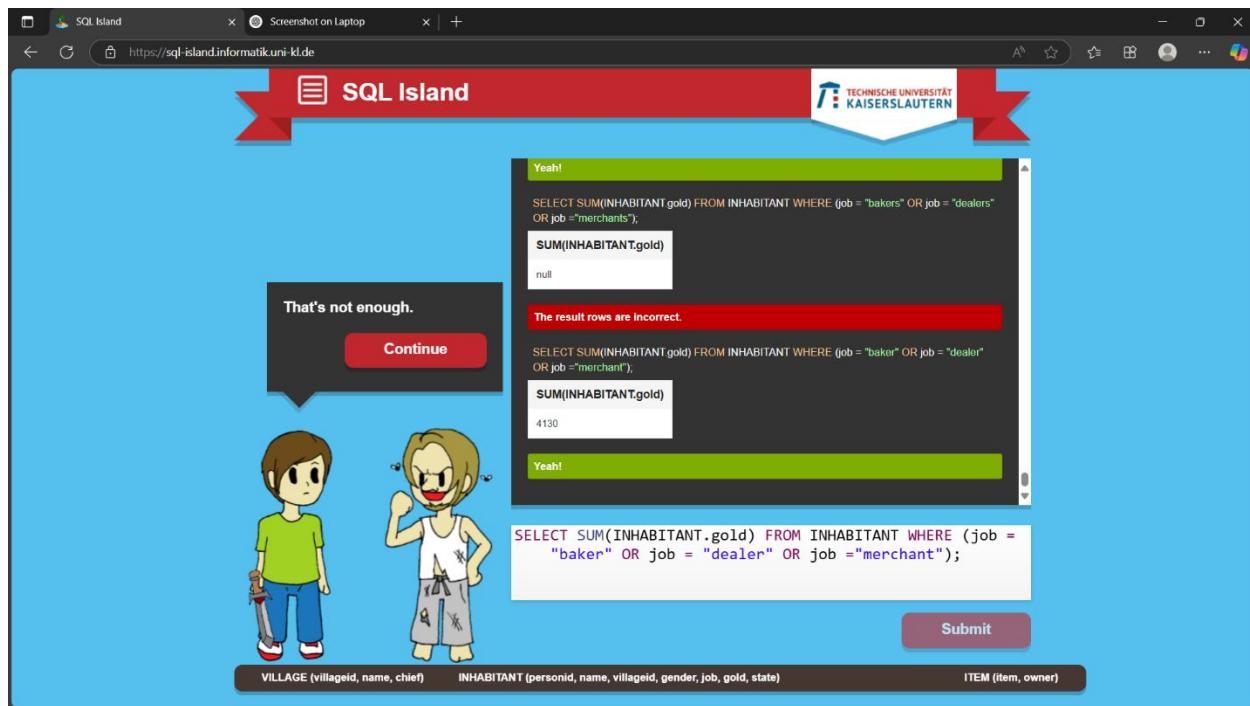
```
SELECT INHABITANT.name FROM INHABITANT, VILLAGE WHERE VILLAGE.villageid = INHABITANT.villageid AND VILLAGE.name = "Onionville" AND gender = "f";
```

 Result: `name
Dirty Diane` Hint: `Yeah!`

At the bottom of the query editor, there's a "Submit" button. Below the editor, there are three tabs: "VILLAGE (villageid, name, chief)", "INHABITANT (personid, name, villageid, gender, job, gold, state)", and "ITEM (item, owner)".

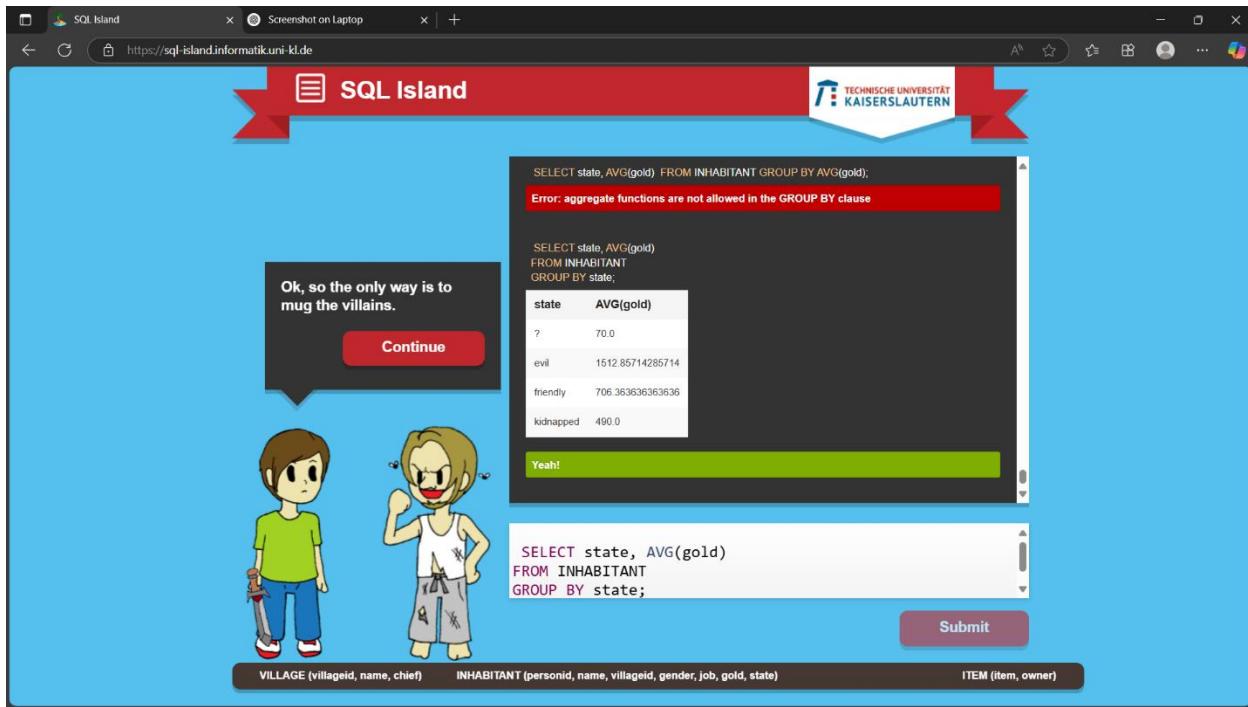
```
SELECT INHABITANT.name FROM INHABITANT, VILLAGE WHERE
VILLAGE.villageid = INHABITANT.villageid AND VILLAGE.name =
"Onionville" AND gender="f";
```

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



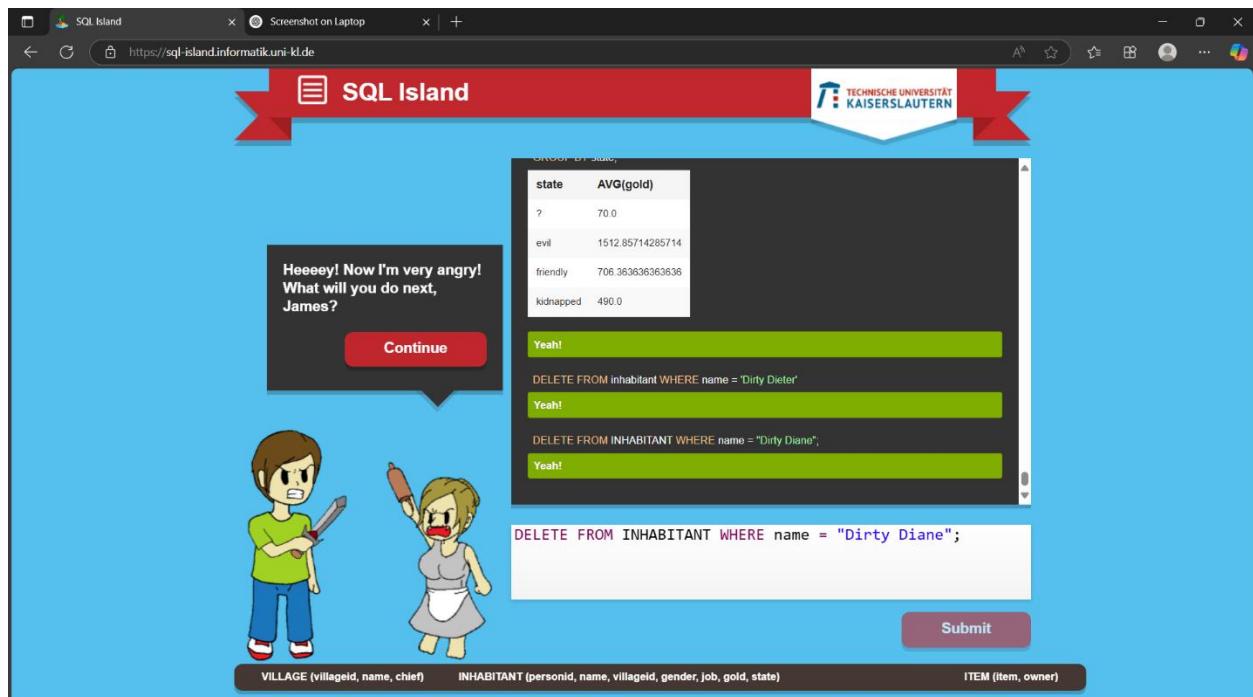
```
SELECT SUM(inhabitant.gold) FROM inhabitant WHERE (job='baker' OR  
job= 'dealer' OR job='merchant');
```

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



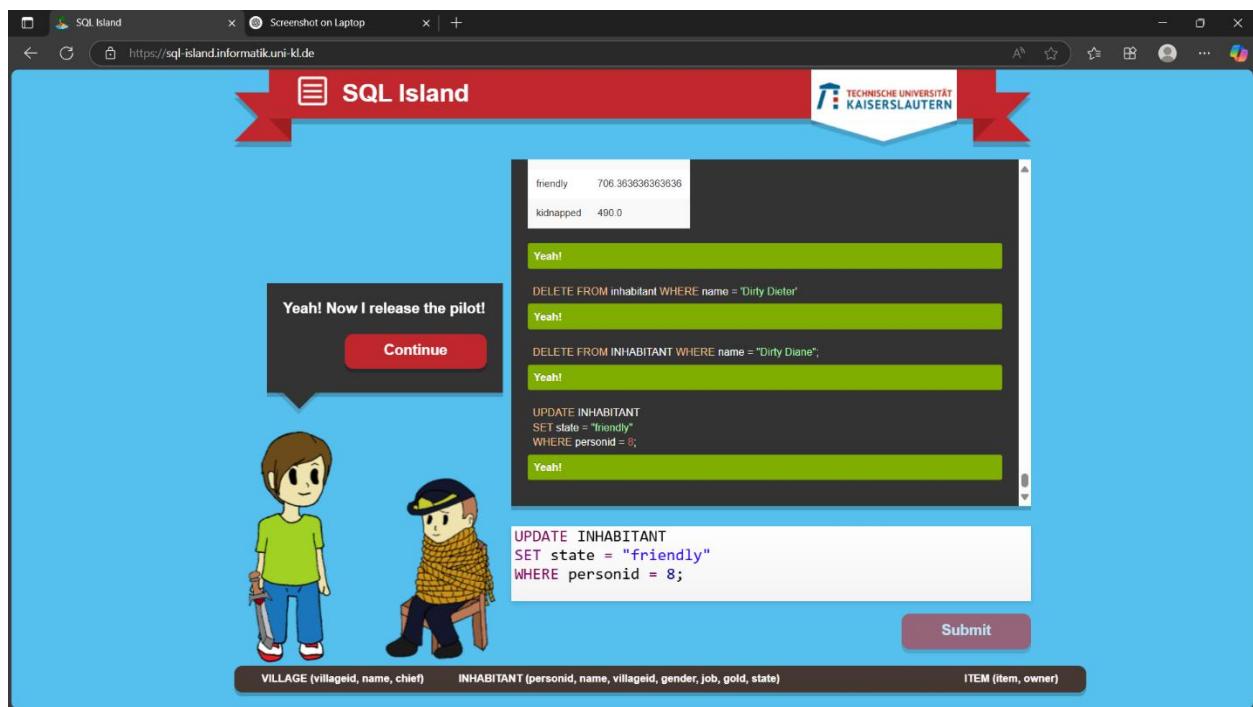
`SELECT state, AVG (gold) FROM inhabitant GROUP BY state ORDER BY AVG (gold);`

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



`DELETE FROM inhabitant WHERE name = 'Dirty Diane';`

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



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'.

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.

Certificate

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;
```

Submit

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