

FINÁLNÍ PROJEKT

č.1



ENGETO

Autor: Denisa Faustková (na discordu Denisa F.)
Datum: 03.09.2024

TESTOVÁNÍ METODY GET

Testovací scénář 1 – Získání dat o existujícím studentovi id 333

1. Otevři MYSQL Workbench: select * from student
2. Otevři Postman, vyber metodu GET, vlož URL:
<http://108.143.193.45:8080/api/v1/students/333>
3. Otevři MYSQL Workbench: select * from student where id = 333
4. Ověř, že vrácená data v Postmanovi souhlasí s daty v databázi studentů
5. Očekávaný výsledek
id = 333; first name = Pepa; last name = OMACKA; email = pepaomacka@gmail.com;
age = 21, status kód = 200 ok

Exekuce testovacího scénáře 1

- Očekávaný a skutečný výsledek je stejný, nebyl nalezen žádný bug, test prošel

The image shows two screenshots. The top screenshot is from Postman, displaying the response of a GET request to the URL `http://108.143.193.45:8080/api/v1/students/333`. The response status is 200 OK, and the body is a JSON object: `{ "id": 333, "firstName": "Pepa", "lastName": "OMACKA", "email": "pepaomacka@gmail.com", "age": 21 }`. The bottom screenshot is from MySQL Workbench, showing the query `select * from student where id = 333` and its results in a table grid.

	id	age	email	first_name	last_name
▶	333	21	pepaomacka@gmail.com	Pepa	OMACKA
*	NULL	NULL	NULL	NULL	NULL

Testovací scénář 2 – Získání dat o existujícím studentovi id 504

1. Otevři MYSQL Workbench: select * from student
2. Otevři Postman, vyber metodu GET, vlož URL:
<http://108.143.193.45:8080/api/v1/students/504>
3. Otevři MYSQL Workbench: select * from student where id = 504
4. Ověř, že vrácená data v Postmanovi souhlasí s daty v databázi studentů
5. Očekávaný výsledek
id = 504; first name = Dan; last name = RUPERT; email = dan@youremail.com;
age = 122, status kód = 200 ok

Exekuce testovacího scénáře 2

- Očekávaný a skutečný výsledek je stejný, nebyl nalezen žádný bug, test prošel

The image shows two screenshots. The top screenshot is from Postman, displaying the response body of a GET request to <http://108.143.193.45:8080/api/v1/students/504>. The status is 200 OK, and the response is a JSON object:

```
{  "id": 504,  "firstName": "Dan",  "lastName": "RUPERT",  "email": "dan@youmail.com",  "age": 122}
```

. The bottom screenshot is from MySQL Workbench, showing the SQL query `select * from student where id = 504` and its result grid. The result grid contains one row with the following data: id: 504, age: 122, email: dan@youmail.com, first_name: Dan, last_name: RUPERT.

id	age	email	first_name	last_name
504	122	dan@youmail.com	Dan	RUPERT
NULL	NULL	NULL	NULL	NULL

Testovací scénář 3 – získání dat o neexistujícím studentovi id 37

1. Otevři MYSQL Workbench: select * from student
2. Otevři Postman, vyber metodu GET, vlož URL:
<http://108.143.193.45:8080/api/v1/students/37>
3. Otevři MYSQL Workbench a zadej příkaz: select * from student where id = 37
Ověř jestli student v databázi opravdu neexistuje
4. Očekávaný výsledek v Postmanovi
Hláška „Student s id 37 neexistuje“

Exekuce testovacího scénáře 3

- Očekávaný výsledek se liší od skutečného výsledku, byl nalezen bug, test neprošel

The screenshot displays two parts of a testing environment. The top part is a REST client interface showing a failed HTTP request. The status bar indicates a "500 Internal Server Error" with a response time of 146 ms and a size of 289 B. The response body, shown in JSON format, contains the following details:

```
{
  "timestamp": "2024-09-01T09:31:32.047+00:00",
  "status": 500,
  "error": "Internal Server Error",
  "message": "",
  "path": "/api/v1/students/37%0A"
}
```

The bottom part of the screenshot shows a database query editor and its results. The query executed is:

```
select * from student where id = 37
```

The results are displayed in a grid format with the following columns: id, age, email, first_name, and last_name. The single row of data shows all fields as NULL.

	id	age	email	first_name	last_name
*	NULL	NULL	NULL	NULL	NULL

TESTOVÁNÍ METODY DELETE

Testovací scénář 4 – smazání dat existující studenta id 349

1. Otevři MYSQL Workbench: select * from student
2. Otevři Postman, vyber metodu DELETE, vlož URL:
<http://108.143.193.45:8080/api/v1/students/349>
3. Otevři MYSQL Workbench: select * from student where id = 349
4. Ověř jestli se student z databáze opravdu vymazal
5. Očekávaný výsledek
Data o studentovi se vymažou z datábase, v Postmanovi status kód 200 ok

Exekuce testovacího scénáře 4

- Očekávaný a skutečný výsledek je stejný, nebyl nalezen žádný bug, test prošel

The image shows two screenshots. The top screenshot is from Postman, displaying a successful DELETE request to the URL <http://108.143.193.45:8080/api/v1/students/349>. The response status is 200 OK, with a response time of 279 ms and a body size of 123 B. The response body is empty, indicating successful deletion.

The bottom screenshot is from MySQL Workbench, showing the SQL query `select * from student where id = 349`. The result grid below the query shows a single row with all fields (id, age, email, first_name, last_name) set to NULL, confirming that the student record has been removed from the database.

	id	age	email	first_name	last_name
*	NULL	NULL	NULL	NULL	NULL

Testovací scénář 5 – smazání dat neexistujícího studenta studenta id 29

1. Otevři MYSQL Workbench: select * from student
2. Otevři Postman, vyber metodu DELETE, vlož URL:
<http://108.143.193.45:8080/api/v1/students/29>
3. Otevři MYSQL Workbench: select * from student where id = 29
4. Ověř zda student v databázi opravdu neexistuje
5. Očekávaný výsledek v Postmanovi
Hláška „Student s id 29 neexistuje, nelze vymazat“

Exekuce testovacího scénáře 5

- Očekávaný a skutečný výsledek se liší, test neprošel, byl nalezen bug

The screenshot displays two application windows. The top window is Postman, showing a DELETE request to the URL `http://108.143.193.45:8080/api/v1/students/29`. The response is a JSON object indicating an error: `{ "timestamp": "2024-09-02T17:47:58.740+00:00", "status": 500, "error": "Internal Server Error", "message": "", "path": "/api/v1/students/29" }`. The bottom window is MySQL Workbench, showing the query `select * from student where id = 29`. The result grid below the query is empty, with all fields (id, age, email, first_name, last_name) showing NULL values.

Postman Response (JSON):

```
{
  "timestamp": "2024-09-02T17:47:58.740+00:00",
  "status": 500,
  "error": "Internal Server Error",
  "message": "",
  "path": "/api/v1/students/29"
}
```

MySQL Query:

```
select * from student where id = 29
```

MySQL Result Grid:

	id	age	email	first_name	last_name
*	NULL	NULL	NULL	NULL	NULL

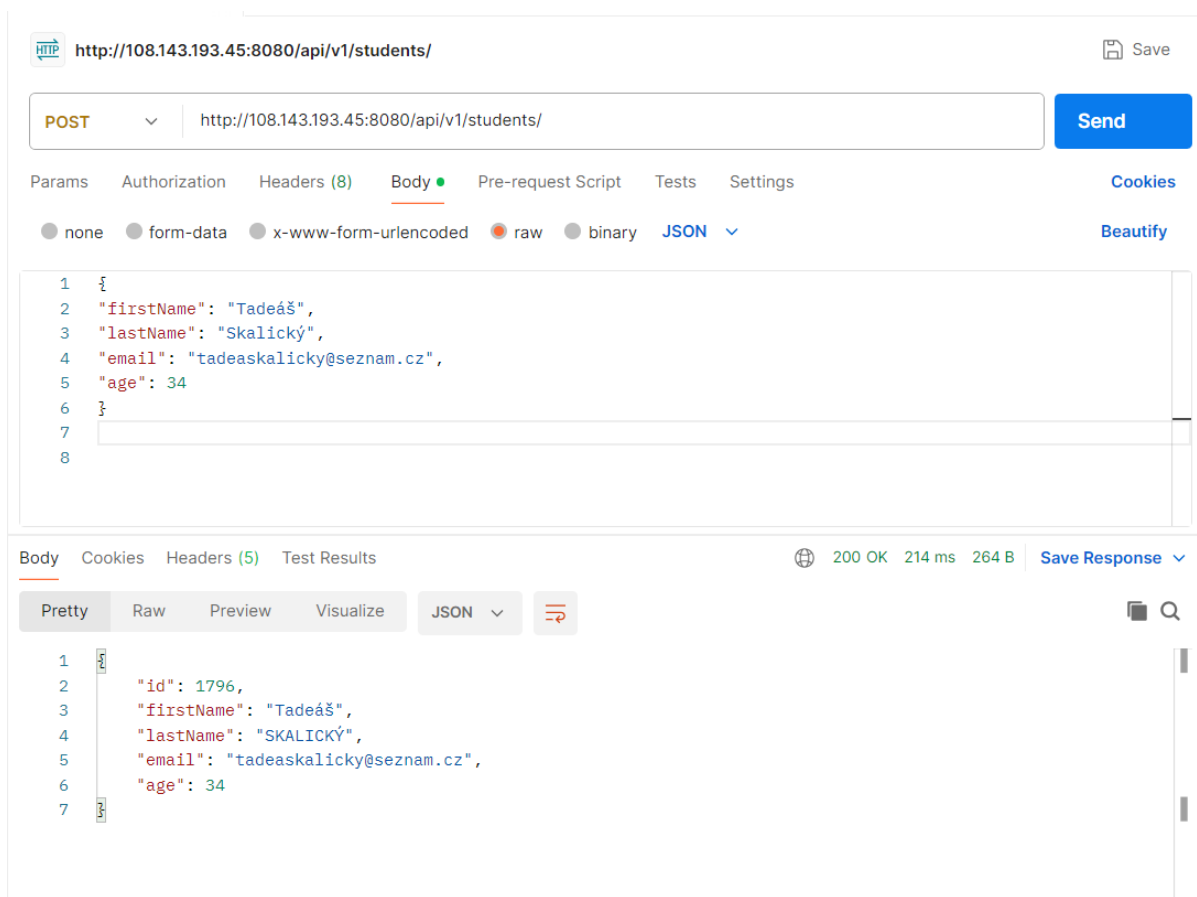
TESTOVÁNÍ METODY POST

Testovací scénář 6 - přidání studenta do databáze

1. Otevři Postman, vyber metodu POST, vlož URL:
<http://108.143.193.45:8080/api/v1/students/>
2. Přepni se do záložky Body, vyber JSON
3. Vlož data o studentovi: „firstName“: „Tadeáš“; „lastName“: „Skalický“;
email: „tadeaskalicky@seznam.cz“; age: 34
4. Otevři MYSQL Workbench: select * from student where id = 1796
5. Ověř, že data v Postmanovi souhlasí s daty v databázi studentů
6. Očekávaný výsledek - id = 1796; first name = Tadeáš; last name = Skalický;
email = tadeaskalicky@seznam.cz; age = 34, status kód = 200 ok

Exekuce testovacího scénáře 6

- Očekávaný a skutečný výsledek se liší, byl nalezen bug, test neprošel



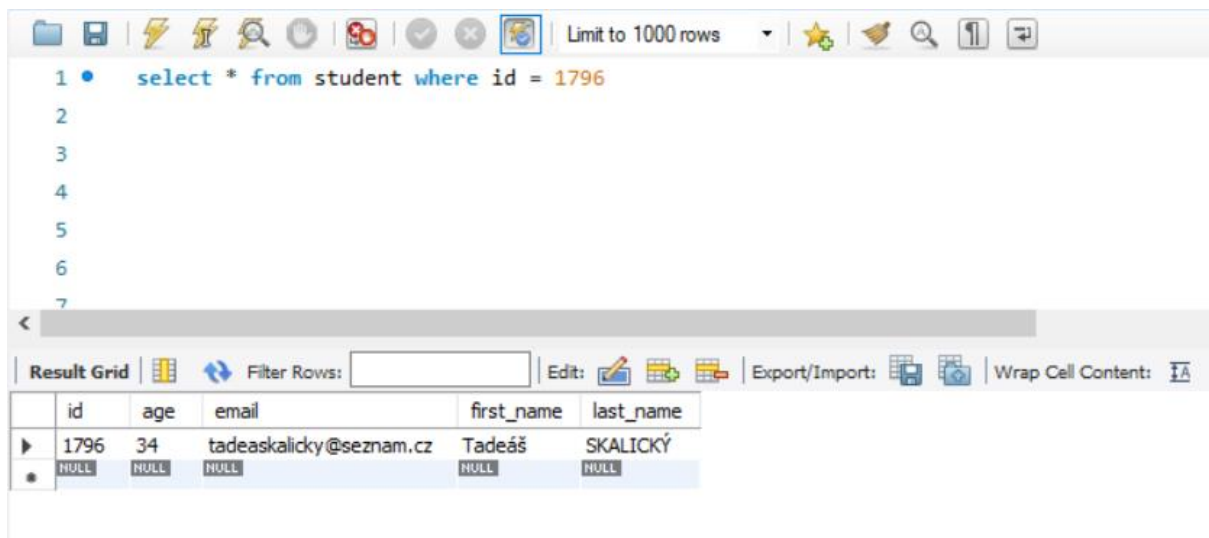
The screenshot shows the Postman interface for a POST request to `http://108.143.193.45:8080/api/v1/students/`. The request body is a JSON object:

```
{
  "firstName": "Tadeáš",
  "lastName": "Skalický",
  "email": "tadeaskalicky@seznam.cz",
  "age": 34
}
```

The response is a 200 OK status with a response time of 214 ms and a body size of 264 B. The response body is a JSON object:

```
{
  "id": 1796,
  "firstName": "Tadeáš",
  "lastName": "SKALICKÝ",
  "email": "tadeaskalicky@seznam.cz",
  "age": 34
}
```

The discrepancy is in the `lastName` field, which is `SKALICKÝ` in the response instead of `Skalický` as expected.



Testovací scénář 7 - přidání studenta do databáze

1. Otevři Postman, vyber metodu POST, vlož URL:
<http://108.143.193.45:8080/api/v1/students/>
2. Přepni se do záložky Body, vyber JSON
3. Vlož data o studentovi: „firstName“: „Adéla“; „lastName“: „Kašpárková“; email: „adelkaspar@gmail.com“; age: 23
4. Otevři MYSQL Workbench: `select * from student where id = 1797`
5. Ověř, že data v Postmanovi souhlasí s daty v databázi studentů
6. Očekávaný výsledek - id = 1797; first name = Adéla; last name = Kašpárková; email = adelkaspar@gmail.com; age = 23, status kód = 200 ok

Exekuce testovacího scénáře 7

- Očekávaný výsledek se liší od skutečného výsledku, byl nalezen bug, test neprošel

Testovací scénář 8 - přidání studenta do databáze

1. Otevři Postman, vyber metodu POST, vlož URL:
<http://108.143.193.45:8080/api/v1/students/>
2. Přepni se do záložky Body, vyber JSON
3. Vlož data o studentovi: „firstName“: „Ester“; „lastName“: „FIALOVÁ“; email: „fialovaster@gmail.com“; age: 19
4. Otevři MYSQL Workbench: select * from student where id = 1798
5. Ověř, že data v Postmanovi souhlasí s daty v databázi studentů
6. Očekávaný výsledek - id = 1798; first name = Ester; last name = FIALOVÁ; email = fialovaster@gmail.com; age = 19, status kód = 200 ok

Exekuce testovacího scénáře 8

- Očekávaný a skutečný výsledek je stejný, nebyl nalezen žádný bug, test prošel

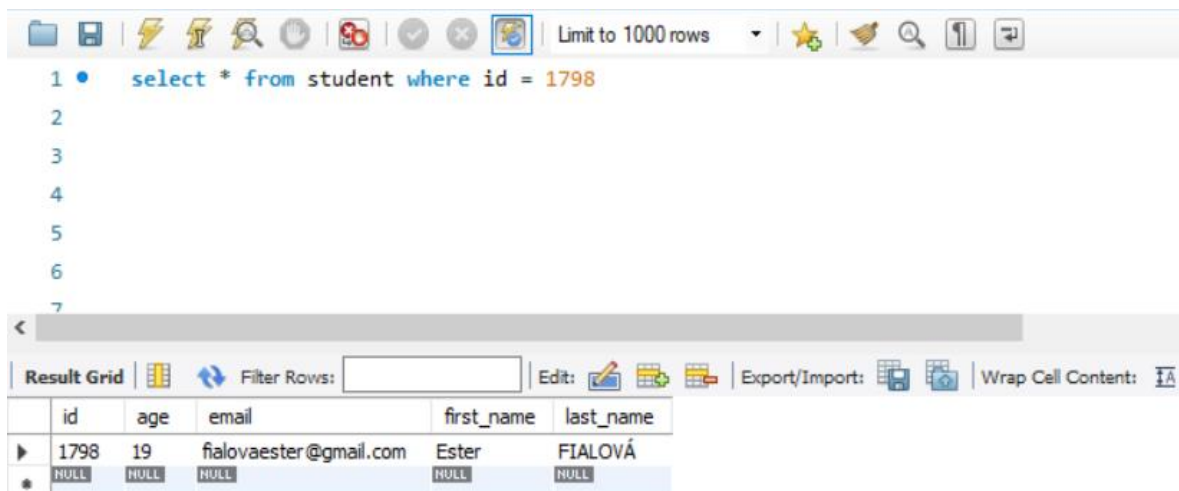
The screenshot displays the Postman interface for a REST client. At the top, the URL bar shows `http://108.143.193.45:8080/api/v1/students/`. The method is set to **POST**. The **Body** tab is selected, showing a JSON payload:

```
1 {
2   "firstName": "Ester",
3   "lastName": "FIALOVÁ",
4   "email": "fialovaester@gmail.com",
5   "age": 19
6 }
```

Below the request, the **Test Results** section shows a successful response:

```
1 {
2   "id": 1798,
3   "firstName": "Ester",
4   "lastName": "FIALOVÁ",
5   "email": "fialovaester@gmail.com",
6   "age": 19
7 }
```

The status bar at the bottom indicates a **200 OK** response with a response time of **217 ms** and a body size of **259 B**.



Testovací scénář 9 - přidání studenta do databáze

1. Otevři Postman, vyber metodu POST, vlož URL:
<http://108.143.193.45:8080/api/v1/students/>
2. Přepni se do záložky Body, vyber JSON
3. Vlož data o studentovi: „firstName“: „LUKAS“; „lastName“: „PETR“;
email: „lukaspetr@centrum.cz“; age: 25
4. Otevři MYSQL Workbench: `select * from student where id = 1801`
5. Ověř, že data v Postmanovi souhlasí s daty v databázi studentů
6. Očekávaný výsledek - id = 1801; first name = LUKAS; last name = PETR;
email = lukaspetr@centrum.cz; age = 25, status kód = 200 ok

Exekuce testovacího scénáře 9

- Očekávaný a skutečný výsledek je stejný, nebyl nalezen žádný bug, test prošel

BUG REPORTY

Bug report 1 – neexistující student id 37

1. Otevři MYSQL Workbench: select * from student
2. Otevři Postman, vyber metodu GET, vlož URL:
http://108.143.193.45:8080/api/v1/students/37
3. Otevři MYSQL Workbench a zadej příkaz: select * from student where id = 37
Ověř zda student v databázi opravdu neexistuje
4. Očekávaný výsledek v Postmanovi
Hláška „Student s id 37 neexistuje“
5. Skutečný výsledek v Postmanovi
Status kód 500 Internal Server Error

The screenshot displays two application windows. The top window is the Postman API client, showing a GET request to the URL `http://108.143.193.45:8080/api/v1/students/37`. The response status is `500 Internal Server Error` with a response time of 146 ms and a body size of 289 B. The response body is shown in JSON format:

```
{  "timestamp": "2024-09-01T09:31:32.047+00:00",  "status": 500,  "error": "Internal Server Error",  "message": "",  "path": "/api/v1/students/37%A"}
```

The bottom window is the MySQL Workbench query editor, showing the SQL query `select * from student where id = 37`. The query results are displayed in a table with the following structure:

id	age	email	first_name	last_name
*	NULL	NULL	NULL	NULL

Bug report 2 – neexistující student id 29

1. Otevři MYSQL Workbench: select * from student
2. Otevři Postman, vyber metodu GET, vlož URL:
http://108.143.193.45:8080/api/v1/students/29
3. Otevři MYSQL Workbench: select * from student where id = 29
4. Očekávaný výsledek v Postmanovi
Hláška „Student s id 29 neexistuje, proto nelze vymazat“
5. Skutečný výsledek v Postmanovi
Status kód 500 Internal Server Error

Body Cookies Headers (4) Test Results 500 Internal Server Error 165 ms 286 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "timestamp": "2024-09-02T18:15:44.149+00:00",
3   "status": 500,
4   "error": "Internal Server Error",
5   "message": "",
6   "path": "/api/v1/students/29"
7 }
```

Limit to 1000 rows

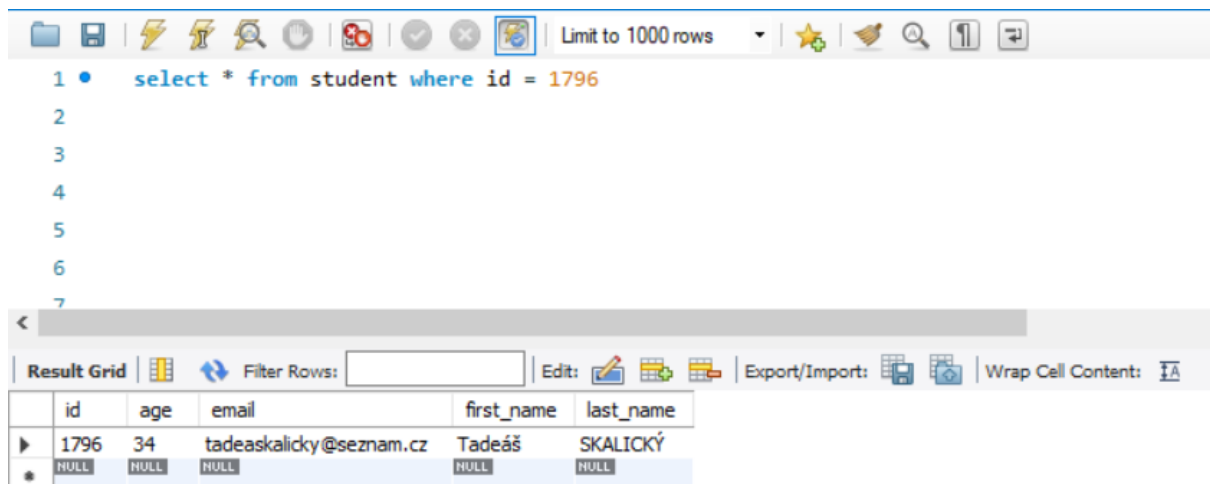
```
1 select * from student where id = 29
2
3
4
5
6
7
```

Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content:

	id	age	email	first_name	last_name
*	NULL	NULL	NULL	NULL	NULL

Bug report 3 – přidání studenta id 1796

1. Otevři Postman, vyber metodu POST, vlož URL:
`http://108.143.193.45:8080/api/v1/students/`
2. Přepni se do záložky Body, vyber JSON
3. Vlož data o studentovi: „firstName“: „Tadeáš“; „lastName“: „Skalický“; email:
„tadeaskalicky@seznam.cz“; age: 34
4. Otevři MYSQL Workbench: `select * from student where id = 1796`
5. Ověř, že data v Postmanovi souhlasí s daty v databázi studentů
6. Očekávaný výsledek: id = 1796; first name = Tadeáš; last name = Skalický; email =
tadeaskalicky@seznam.cz; age = 34, status kód = 200 ok
7. Skutečný výsledek: id = 1796; first name = Adéla; last name = SKALICKÝ; email =
tadeaskalicky@seznam.cz; age = 34



1 • `select * from student where id = 1796`

2

3

4

5

6

7

Result Grid

Filter Rows:

Edit:

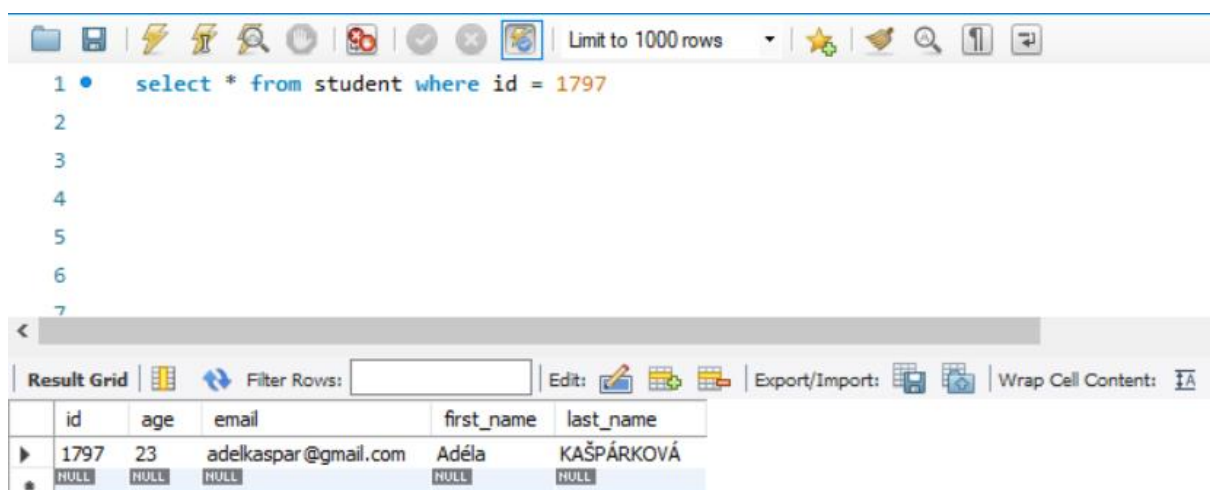
Export/Import:

Wrap Cell Content:

	id	age	email	first_name	last_name
▶	1796	34	tadeaskalicky@seznam.cz	Tadeáš	SKALICKÝ
*	NULL	NULL	NULL	NULL	NULL

Bug report 4 – přidání studenta id 1797

1. Otevři Postman, vyber metodu POST, vlož URL:
`http://108.143.193.45:8080/api/v1/students/`
2. Přepni se do záložky Body, vyber JSON
3. Vlož data o studentovi: „firstName“: „Adéla“; „lastName“: „Kašpárková“; email:
„adelkaspar@gmail.com“; age: 23
4. Otevři MYSQL Workbench: `select * from student where id = 1797`
5. Ověř, že data v Postmanovi souhlasí s daty v databázi studentů
6. Očekávaný výsledek: id = 1797; first name = Adéla; last name = Kašpárková;
email = adelkaspar@gmail.com; age = 23, status kód = 200 ok
7. Skutečný výsledek: id = 1797; first name = Adéla; last name = KAŠPÁRKOVÁ;
email = adelkaspar@gmail.com; age = 23, status kód = 200 ok



1 • `select * from student where id = 1797`

2

3

4

5

6

7

Result Grid

Filter Rows:

Edit:

Export/Import:

Wrap Cell Content:

	id	age	email	first_name	last_name
▶	1797	23	adelkaspar@gmail.com	Adéla	KAŠPÁRKOVÁ
*	NULL	NULL	NULL	NULL	NULL