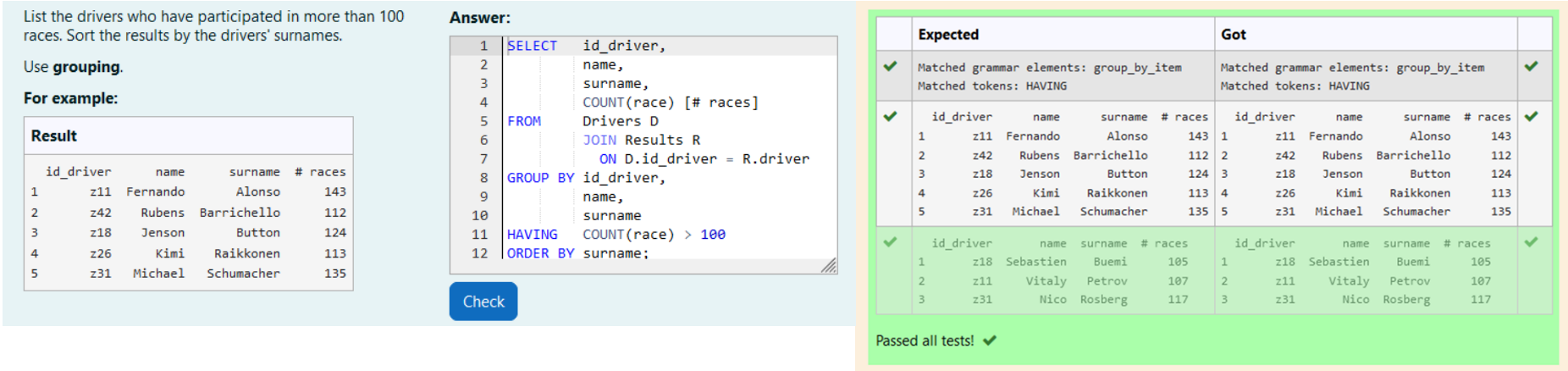
GRANT Privileges, REVOKE Risk: Safe and Scalable Teaching of Database Administration with Isolated Containers

Andrzej Wójtowicz, Maciej Prill

Adam Mickiewicz University, Poznań, Poland

Background

- SQL in “Introduction to databases” courses (DQL, DDL, DML)
- Autograders for LMS (Moodle) supporting other engines than SQLite:
  - A. Wójtowicz et al. (2025, SIGCSE). *Relational Database Courses with CodeRunner in Moodle: Extending SQL Programming Assignments to Client-Server Database Engines.*
  - L. Behme et al. (2025, DataEd). *Teaching Large-Scale Data Management to Large Cohorts of Undergraduate Students.*



List the drivers who have participated in more than 100 races. Sort the results by the drivers' surnames.

Use **grouping**.

For example:

Result

	id_driver	name	surname	# races
1	z11	Fernando	Alonso	143
2	z42	Rubens	Barrichello	112
3	z18	Jenson	Button	124
4	z26	Kimi	Raikkonen	113
5	z31	Michael	Schumacher	135

Answer:

```
1  SELECT   id_driver,
2           name,
3           surname,
4           COUNT(race) [# races]
5  FROM     Drivers D
6           JOIN Results R
7             ON D.id_driver = R.driver
8  GROUP BY id_driver,
9           name,
10          surname
11 HAVING   COUNT(race) > 100
12 ORDER BY surname;
```

Check

	Expected	Got	
✔	Matched grammar elements: group_by_item Matched tokens: HAVING	Matched grammar elements: group_by_item Matched tokens: HAVING	✔
✔	id_driver name surname # races 1 z11 Fernando Alonso 143 2 z42 Rubens Barrichello 112 3 z18 Jenson Button 124 4 z26 Kimi Raikkonen 113 5 z31 Michael Schumacher 135	id_driver name surname # races 1 z11 Fernando Alonso 143 2 z42 Rubens Barrichello 112 3 z18 Jenson Button 124 4 z26 Kimi Raikkonen 113 5 z31 Michael Schumacher 135	✔
✔	id_driver name surname # races 1 z18 Sebastien Buemi 105 2 z11 Vitaly Petrov 107 3 z31 Nico Rosberg 117	id_driver name surname # races 1 z18 Sebastien Buemi 105 2 z11 Vitaly Petrov 107 3 z31 Nico Rosberg 117	✔

Passed all tests! ✔

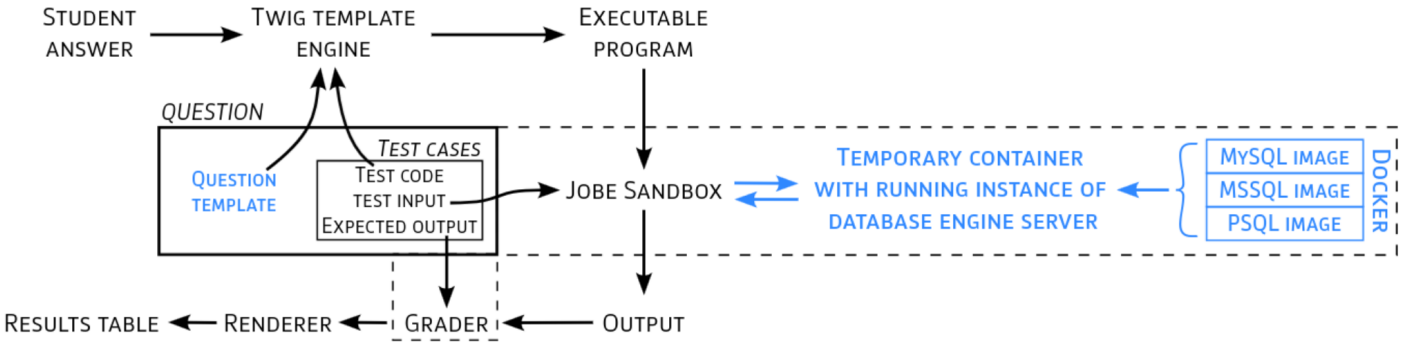
Motivation

- Next course after “Introduction to databases” – survey (*n*=36, multiple choice):
  - 50% – database administration,
  - 39% – query optimization,
  - 31% – database engine implementation,
  - 19% – data warehouses.

- A lot of attention being paid to cybersecurity (OECD, the U.S. Department of Defense, the U.K. Department for Work and Pensions).

Our solution – technical part

- Stick to Moodle LMS and Jobe/CodeRunner autograder.
- Focus on MSSQL, PostgreSQL and MySQL.
- On-demand starting short-lived Docker containers with customized full database engine servers.



Our solution – example – DCL task definition

Question text

```
Grant the user `andy@%` privileges for the `SELECT`, `DROP`, and
`INSERT` operations on the `animals` table in the `smart_data` db.
```

Answer

```
GRANT SELECT, DROP, INSERT ON smart_data.animals TO 'andy'@'%';
```

Test case 1

Expected output

	User	Host	Db	Table_name	Select	Drop	Insert
1	andy	%	smart data	animals	true	true	true

Extra template data

```
__db_prepare__ = """
CREATE DATABASE smart_data; USE smart_data;
CREATE TABLE animals (
  id      INT PRIMARY KEY,
  name    VARCHAR(20),
  owner   VARCHAR(20),
  species VARCHAR(20),
  sex     CHAR(1));
INSERT INTO animals VALUES
(4, 'Fluffy', 'Harold', 'cat', 'f'),
(5, 'Claws' , 'Gwen'  , 'cat', 'm'),
(9, 'Buffy' , 'Harold', 'dog', NULL);
CREATE USER 'andy'@'%' IDENTIFIED BY 'pazzw0Rd';
"""
invoke_cursor_sql(__db_prepare__, database=None)
invoke_cursor_sql(__student_answer__, database="smart_data")
__sql_verify__ = """
SELECT User, Host, Db, Table_name,
  CASE WHEN FIND_IN_SET('Select', Table_priv) > 0
       THEN 'true' ELSE 'false' END AS 'Select',
  CASE WHEN FIND_IN_SET('Drop', Table_priv) > 0
       THEN 'true' ELSE 'false' END AS 'Drop',
  CASE WHEN FIND_IN_SET('Insert', Table_priv) > 0
       THEN 'true' ELSE 'false' END AS 'Insert'
FROM   mysql.tables_priv
WHERE  User = 'andy' AND Host = '%';
"""
print(invoke_cursor_sql(__sql_verify__, database=None))
```

Our solution – example – DCL task run

Grant the user andy@% the privileges to perform SELECT, DROP, and INSERT operations on the animals table in the smart_data db.

Answer:

```
GRANT SELECT, DROP, INSERT
   ON smart_data.animals
   TO 'andy'@'%';
```

Check

	Expected							Got							
✓		User	Host	Db	Table_name	Select	Drop	Insert		User	Host	Db	Table_name	Select	Drop	Insert	✓
	1	andy	%	smart_data	animals	true	true	true	1	andy	%	smart_data	animals	true	true	true	

…it works also with Bash commands tasks, e.g., *mysqlpump*

Evaluation

- 15 weeks course, 17 students.
- 2h lab/week.
- Final grade: 75% autograder, 25% project.
- 3 database engines: MySQL, Microsoft SQL Server, and PostgreSQL.
- Topics covered:
  - (1) server installation and client tools;
  - (2) users, roles, and permissions management;
  - (3) password and data schema management;
  - (4) file operations, backups and recovery;
  - (5) data partitioning.
- VMs for practice.
- Jobe server: 16 CPUs, 64 GB RAM.

Evaluation – continued

- Post-course survey (statistics in the paper):
  - students were generally satisfied with the course structure and content,
  - subject of the classes was rated as interesting and useful,
  - difficulty level of the autograder questions was rated as moderate,
  - positive opinions about integrating three different database engines.

- CodeRunner/Jobe environment:
  - nothing crashed 😉,
  - Microsoft SQL Server container execution ~2x longer than two other database engines (avg 10.0 s vs 4.0 s),
  - students' subjective assessment of the autograder's speed perceived as balanced.

Limitations and further work

- Support for additional database engines (e.g., Oracle).
- No automated, context-aware hints (e.g., integrate syntax analysis and RAG-enabled LLMs).
- No partial grading for complex tasks.
- No support for long-running tasks or multi-server scenarios.
- No realistic DBA scenarios (add fault-injection and adversarial user simulations).

Thank you!

https://github.com/andre-wojtowicz/coderunner-dba-dataed2026