ECE30030/ITP30010 Database Systems

Handshaking with an R-DBMS

Charmgil Hong

charmgil@handong.edu

Spring, 2023

Handong Global University

Last Lecture: Relational Algebra

- A procedural language consisting of a set of operations that take one or two relations as input and produce a new relation as their output

- Basic operators
  - Select: σ
  - Project: ∏
  - Cartesian product: ×
  - Join: ⋈
  - Rename: ρ

  - Union: ∪
  - Set-intersection: ∩
  - Set-difference: –

Last Lecture: Select Operation

- The select operation selects tuples that satisfy a given predicate
- Notation: $\sigma_p(r)$
  - p is called the selection predicate

- Example: select those tuples of the instructor relation where the instructor is in the "Comp. Sci." department
  - Query: $\sigma_{\text{dept\_name}=\text{“Comp. Sci.”}}(\text{instructor})$

  - Result

ID	name	dept_name	salary
10101	Srinivasan	Comp. Sci.	65000.00
45565	Katz	Comp. Sci.	75000.00
83821	Brandt	Comp. Sci.	92000.00

Last Lecture: Select Operation

- Comparisons using $=, \neq, >, \geq, <, \leq$ are allowed in the selection predicates
- Combine several predicates into a larger predicate using the connectives: $\wedge$ (**and**), $\vee$ (**or**), $\neg$ (**not**)

- Example: Find the instructors in Comp. Sci. with a salary greater than $70,000
  - Query: $\sigma_{\text{dept\_name}=\text{“Comp. Sci.”} \wedge \text{salary}>70{,}000}(\text{instructor})$

  - Result

ID	name	dept_name	salary
45565	Katz	Comp. Sci.	75000.00
83821	Brandt	Comp. Sci.	92000.00

Last Lecture: Project Operation

- Example: eliminate the ID and dept_name attributes of instructor
  - Query: $\Pi_{name,\ salary}(instructor)$
  - Result:

Projected relation

name	salary
Srinivasan	65000.00
Wu	90000.00
Mozart	40000.00
Einstein	95000.00
El Said	60000.00
Gold	87000.00
Katz	75000.00
Califieri	62000.00
Singh	80000.00
Crick	72000.00
Brandt	92000.00
Kim	80000.00

Original relation

ID	name	dept_name	salary
10101	Srinivasan	Comp. Sci.	65000.00
12121	Wu	Finance	90000.00
15151	Mozart	Music	40000.00
22222	Einstein	Physics	95000.00
32343	El Said	History	60000.00
33456	Gold	Physics	87000.00
45565	Katz	Comp. Sci.	75000.00
58583	Califieri	History	62000.00
76543	Singh	Finance	80000.00
76766	Crick	Biology	72000.00
83821	Brandt	Comp. Sci.	92000.00
98345	Kim	Elec. Eng.	80000.00

Last Lecture: *instructor* × *teaches*

- Example: the Cartesian product of the relations *instructor* and t*eaches*
  - Result (total 180 tuples = 12 instructors × 15 courses)

instructor.ID	name	dept_name	salary	teaches.ID	course_id	sec_id	semester	year
10101	Srinivasan	Comp. Sci.	65000	76766	BIO-101	1	Summer	2017
12121	Wu	Finance	90000	76766	BIO-101	1	Summer	2017
15151	Mozart	Music	40000	76766	BIO-101	1	Summer	2017
22222	Einstein	Physics	95000	76766	BIO-101	1	Summer	2017
32343	El Said	History	60000	76766	BIO-101	1	Summer	2017
…	…	…	…	…	…	…	…	…
10101	Srinivasan	Comp. Sci.	65000	10101	CS-101	1	Fall	2017
12121	Wu	Finance	90000	10101	CS-101	1	Fall	2017
15151	Mozart	Music	40000	10101	CS-101	1	Fall	2017
22222	Einstein	Physics	95000	10101	CS-101	1	Fall	2017
32343	El Said	History	60000	10101	CS-101	1	Fall	2017
…	…	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…	…
10101	Srinivasan	Comp. Sci.	65000	83821	CS-190	2	Spring	2017
12121	Wu	Finance	90000	83821	CS-190	2	Spring	2017
15151	Mozart	Music	40000	83821	CS-190	2	Spring	2017
…	…	…	…	…	…	…	…	…
10101	Srinivasan	Comp. Sci.	65000	10101	CS-315	1	Spring	2018
12121	Wu	Finance	90000	10101	CS-315	1	Spring	2018
15151	Mozart	Music	40000	10101	CS-315	1	Spring	2018
…	…	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…	…

Last Lecture: Join Operation

- Example: Get only those tuples of “instructor × teaches” that pertain to the courses that the instructor taught
  - Result

instructor.ID	name	dept_name	salary	teaches.ID	course_id	sec_id	semester	year
10101	Srinivasan	Comp. Sci.	65000	10101	CS-101	1	Fall	2017
10101	Srinivasan	Comp. Sci.	65000	10101	CS-315	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	10101	CS-347	1	Fall	2017
12121	Wu	Finance	90000	12121	FIN-201	1	Spring	2018
15151	Mozart	Music	40000	15151	MU-199	1	Spring	2018
22222	Einstein	Physics	95000	22222	PHY-101	1	Fall	2017
32343	El Said	History	60000	32343	HIS-351	1	Spring	2018
45565	Katz	Comp. Sci.	75000	45565	CS-101	1	Spring	2018
45565	Katz	Comp. Sci.	75000	45565	CS-319	1	Spring	2018
76766	Crick	Biology	72000	76766	BIO-101	1	Summer	2017
76766	Crick	Biology	72000	76766	BIO-301	1	Summer	2018
83821	Brandt	Comp. Sci.	92000	83821	CS-190	1	Spring	2017
83821	Brandt	Comp. Sci.	92000	83821	CS-190	2	Spring	2017
83821	Brandt	Comp. Sci.	92000	83821	CS-319	2	Spring	2018
98345	Kim	Elec. Eng.	80000	98345	EE-181	1	Spring	2017

Last Lecture: Union Operation

- The union operation combines two relations as a superset of both
  - Notation: $r \cup s$

- For $r \cup s$ to be valid,
  1. r, s must have the *same* number of attributes (same arity)
  2. The attribute domains must be compatible
     - *E.g.*, the 2nd column of r deals with the same type of values as does the 2nd column of s

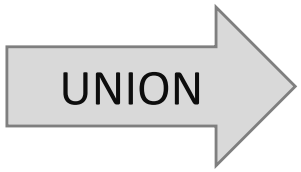
- Example: Find all courses taught in the Fall 2017 semester, or in the Spring 2018 semester, or in both
  - Query: $\prod_{course\_id} (\sigma_{semester=\text{“Fall”} \wedge year=2017} (teaches)) \cup$
    $\prod_{course\_id} (\sigma_{semester=\text{“Spring”} \wedge year=2018} (teaches))$

Last Lecture: Union Operation

- Example: Find all courses taught in the Fall 2017 semester, or in the Spring 2018 semester, or in both
  - Result

course_id
CS-101
CS-347
PHY-101

course_id
CS-101
CS-315
CS-319
FIN-201
HIS-351
MU-199



course_id
CS-101
CS-347
PHY-101
CS-315
CS-319
FIN-201
HIS-351
MU-199

Last Lecture: Set-Intersection Operation

- The set-intersection operation finds tuples that are in both the input relations
  - Notation: $r \cap s$
  - Assumptions:
    - r, s have the same arity
    - Attributes of r and s are compatible

- Example: Find the set of all courses taught in both the 2017-Fall and 2018-Spring semesters
  - Query: $\prod_{course\_id} (\sigma_{semester=\text{“Fall”} \wedge year=2017} (teaches)) \cap \prod_{course\_id} (\sigma_{semester=\text{“Spring”} \wedge year=2018} (teaches))$
  - Result

course_id
CS-101

Last Lecture: Set-Difference Operation

- The set-difference operation finds tuples that are in one relation but are not in another
  - Notation: $r - s$
  - Assumptions:
    - r, s have the same arity
    - Attributes of r and s are compatible

- Example: Find all courses taught in the 2017-Fall semester, but not in the 2018-Spring semester
  - Query: $\prod_{course\_id}(\sigma_{semester=\text{“Fall”} \wedge year=2017}(teaches)) - \prod_{course\_id}(\sigma_{semester=\text{“Spring”} \wedge year=2018}(teaches))$

course_id
CS-347
PHY-101

Last Lecture: Rename Operation

- The results of relational-algebra expressions do not have a name that one can use to refer to them

- The rename operator, ρ, sets names to relational-algebra expressions
  - Notation: $\rho_x(E)$
    - Returns the result of expression *E* under the name *x*

Last Lecture: Equivalent Queries

- There is more than one way to write a query in relational algebra

- Example: Find information about courses taught by instructors in the Comp. Sci. department with salary greater than 50,000
  - Query 1:

    $\sigma_{dept\_name=\text{“Comp. Sci.”} \wedge salary > 50{,}000}(instructor)$

  - Query 2:

    $\sigma_{dept\_name=\text{“Comp. Sci.”}}(\sigma_{salary > 50.000}(instructor))$

  - The two queries are not identical; they are, however, equivalent -- they give the same result on any database

Agenda

- Introduction to MySQL
- SQL preview

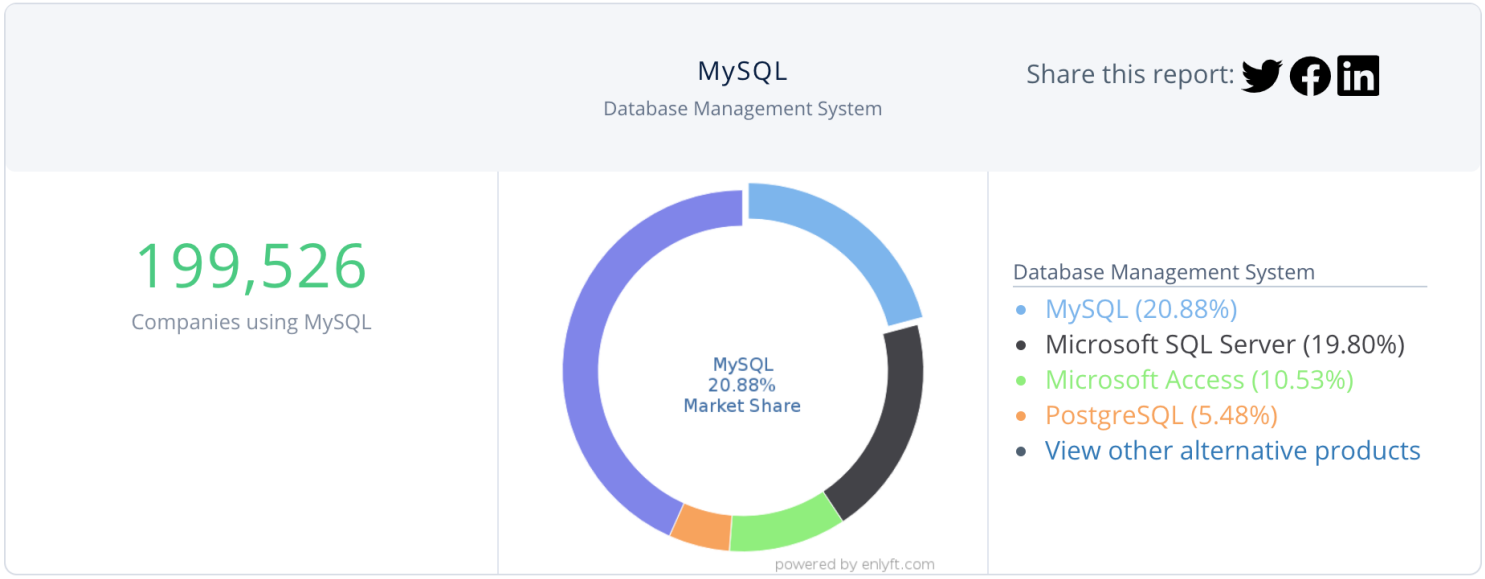
MySQL

- MySQL is an SQL-based relational database management system (DBMS)
  - Free and open-source R-DBMS (under GPL)
    - Owned by Oracle
    - Commercial version of MySQL is also provided (including technical support)
    - "My" came from the name of co-founder Michael Widenius' daughter
      - *C.f.*, MariaDB
  - Compatible with standard SQL
  - Frequently used for commercial web services

* Image src: https://en.wikipedia.org/wiki/MySQL

MySQL

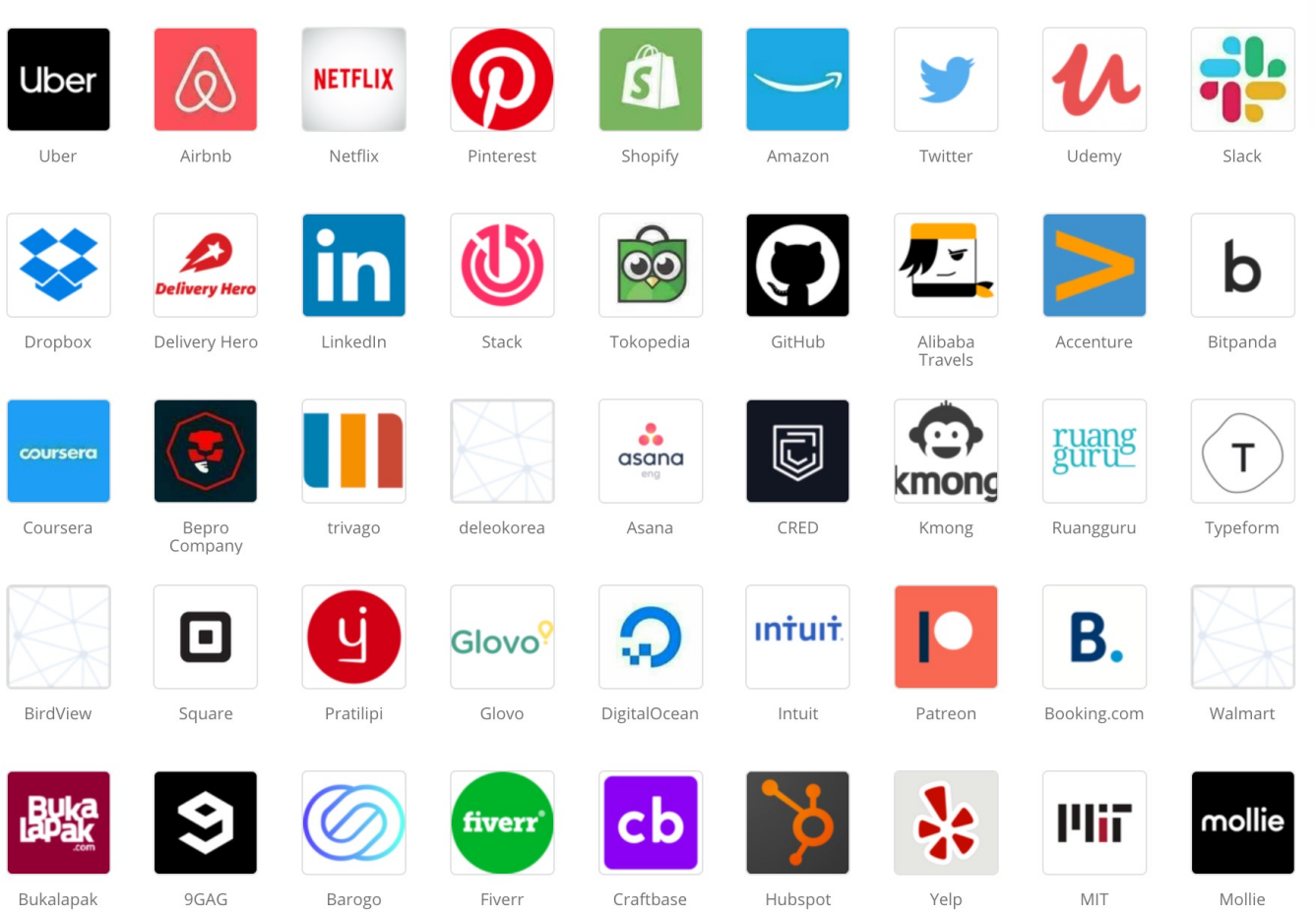
- Companies using MySQL (a study by Enlyft)
  - *"We have data on 199,526 (out of 955,547) companies that use MySQL"*
  - Often used by companies with 10-50 employees and 1M-10M dollars in revenue
    - *C.f.*, Oracle 12 is most often used by companies with 50-200 employees and >1000M dollars in revenue



* Source: https://enlyft.com/tech/products/mysql

MySQL

- Companies using MySQL (full list: https://www.mysql.com/customers/)



* Source: https://stackshare.io/mysql

MySQL

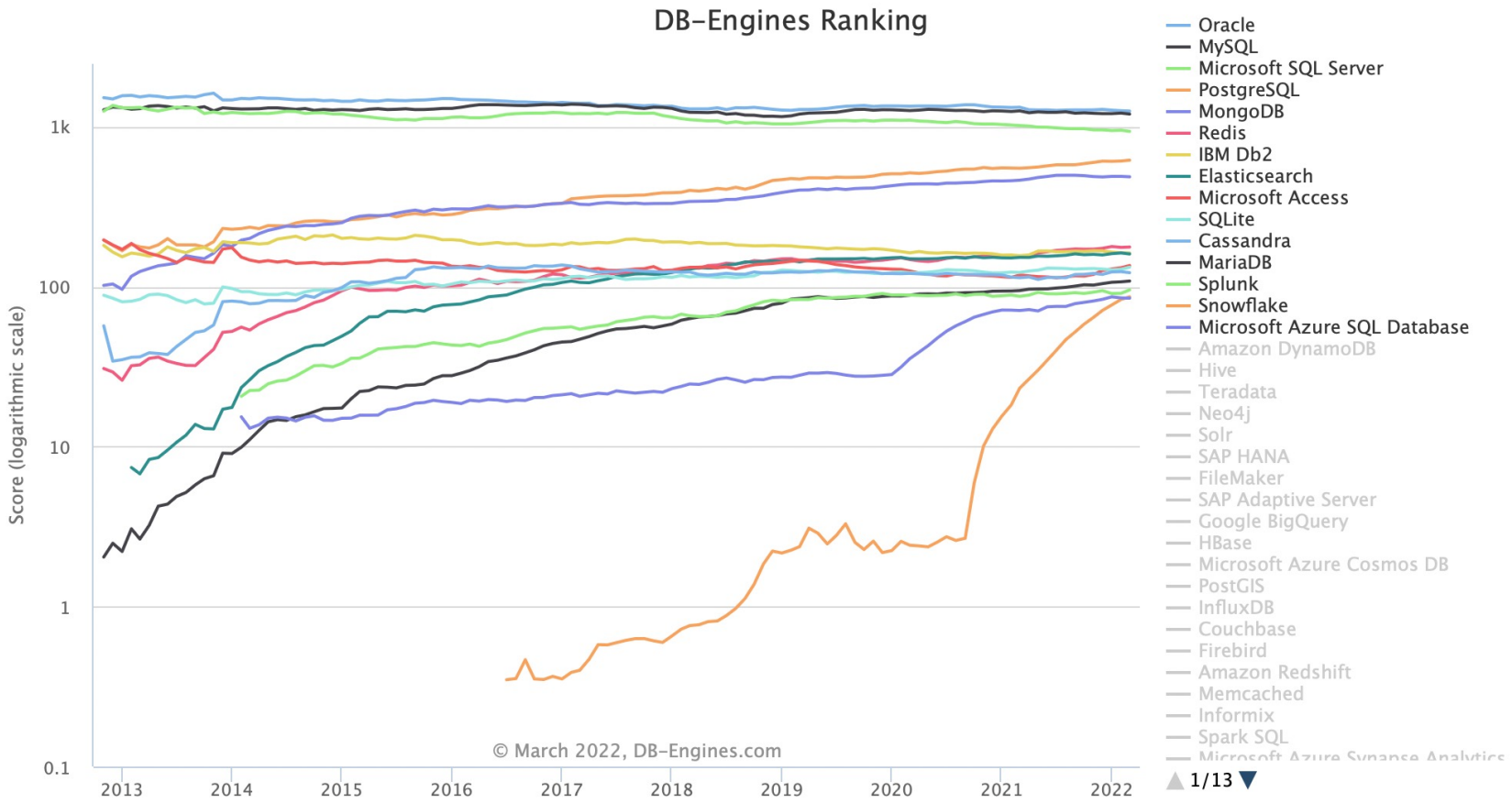
- Why MySQL?
  - Popular
    - Active discussions all over the Internet
  - Versatile: runs on Linux, Windows, Mac OS X, Solaris, FreeBSD, …
    - Supports wide range of programming languages (C/C++, Java, Python, .Net, …)
  - Cost starts from zero
  - High performance (fast and reliable)

388 systems in ranking, March 2022

Rank			DBMS	Database Model	Score		
Mar 2022	Feb 2022	Mar 2021			Mar 2022	Feb 2022	Mar 2021
1.	1.	1.	Oracle	Relational, Multi-model	1251.32	-5.51	-70.42
2.	2.	2.	MySQL	Relational, Multi-model	1198.23	-16.45	-56.59
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	933.78	-15.27	-81.52
4.	4.	4.	PostgreSQL	Relational, Multi-model	616.93	+7.54	+67.64
5.	5.	5.	MongoDB	Document, Multi-model	485.66	-2.98	+23.27
6.	6.	↑ 7.	Redis	Key-value, Multi-model	176.76	+0.96	+22.61
7.	7.	↓ 6.	IBM Db2	Relational, Multi-model	162.15	-0.73	+6.14
8.	8.	8.	Elasticsearch	Search engine, Multi-model	159.95	-2.35	+7.61
9.	9.	↑ 10.	Microsoft Access	Relational	135.43	+4.17	+17.29
10.	10.	↓ 9.	SQLite	Relational	132.18	+3.81	+9.54

* Image src: https://db-engines.com/en/ranking

DBMS Trend Popularity

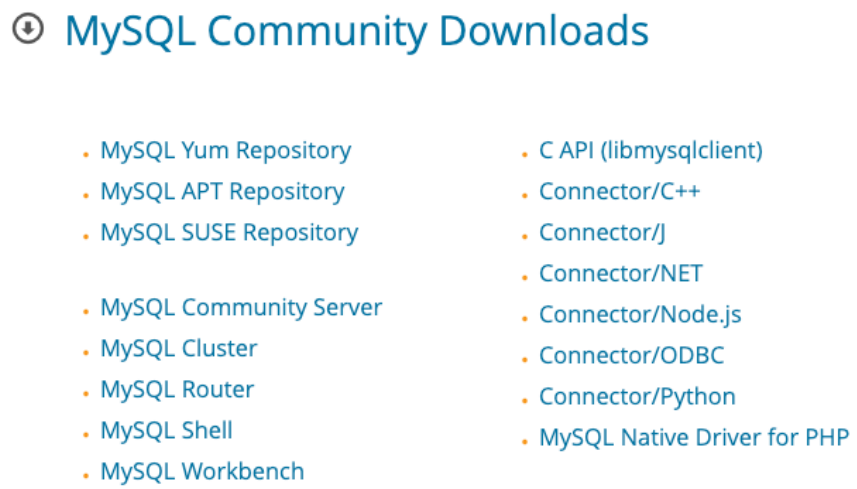


- Score definition (if you are interested): https://db-engines.com/en/ranking_definition

* Image src: https://db-engines.com/en/ranking_trend

MySQL

- Why MySQL?
  - Popular
    - Active discussions all over the Internet
  - Versatile: runs on Linux, Windows, Mac OS X, Solaris, FreeBSD, …
    - Supports wide range of programming languages (C/C++, Java, Python, .Net, …)
  - Cost starts from zero
  - High performance (fast and reliable)



* Image src: https://dev.mysql.com/downloads/

MySQL

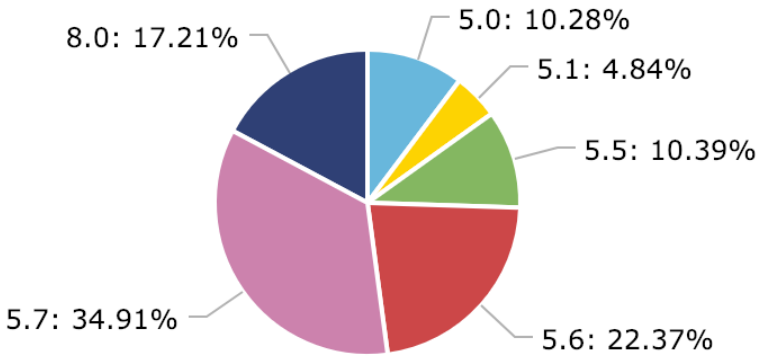
- Massive — Can handle terabytes of data
- Convenient — Supports high-level query language
- Multi-user — Supports concurrent data access
- Safe — Supports transactions
- Efficient — Can handle thousands of queries/second
- Reliable — 99.99% up-time in many real-world products

MySQL Versions

- MySQL 5.x vs 8.0

Version 5.x
- Most popular version of MySQL
- More stable and conventional

Version 8.0
- Current version
- Provides up-to-date DB functionalities (better storage engine, faster, more secure)



* Source: https://www.eversql.com/mysql-8-adoption-usage-rate/#:~:text=MySQL%205.7%20is%20still%20the,17%25%20are%20using%20MySQL%208.

MySQL Versions

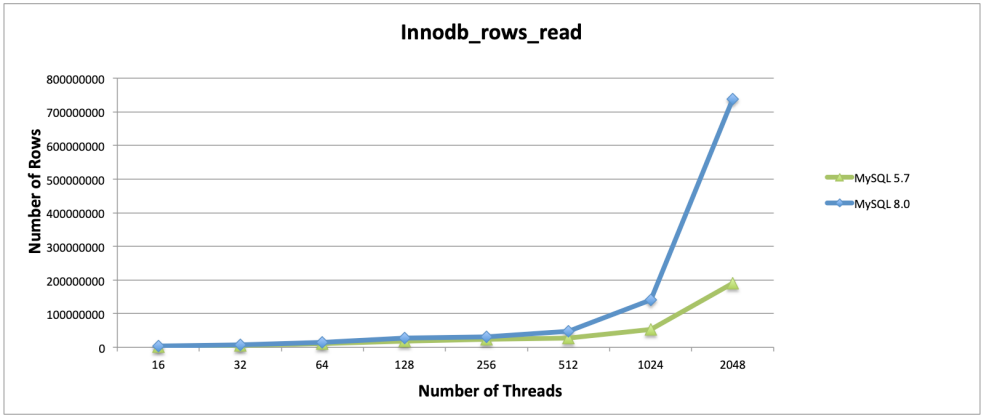
- MySQL 5.x vs 8.0

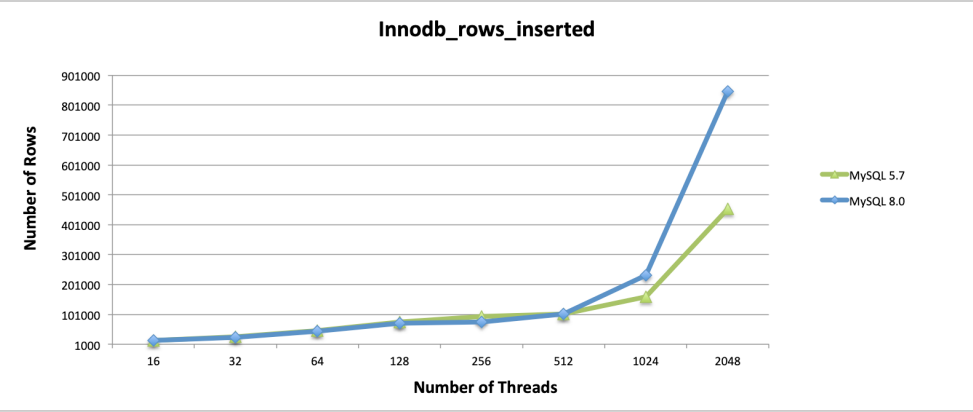
Version 5.x

- Most popular version of MySQL
- More stable and conventional

Version 8.0

- Current version
- Provides up-to-date DB functionalities (better storage engine, faster, more secure)





* Source: https://severalnines.com/database-blog/mysql-performance-benchmarking-mysql-57-vs-mysql-80

MySQL Versions

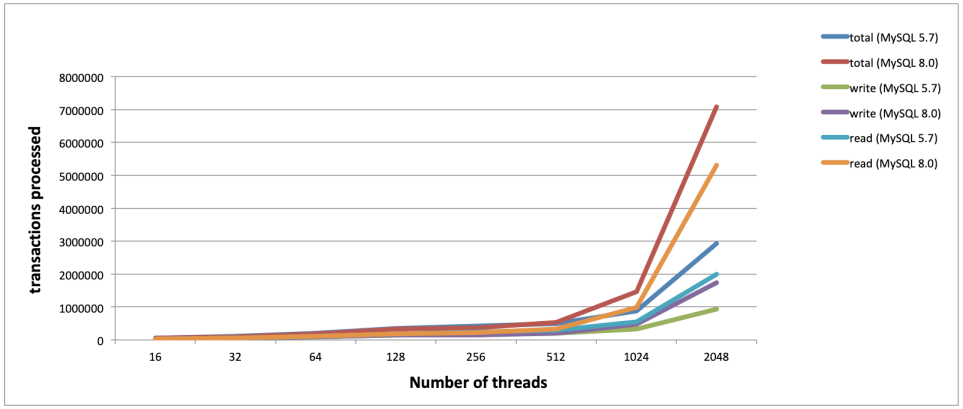
- MySQL 5.x vs 8.0

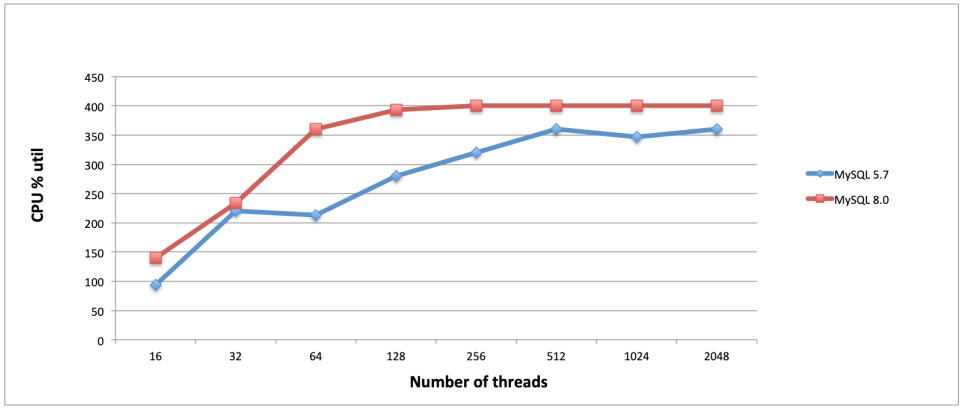
Version 5.x

- Most popular version of MySQL
- More stable and conventional

Version 8.0

- Current version
- Provides up-to-date DB functionalities (better storage engine, faster, more secure)





* Source: https://severalnines.com/database-blog/mysql-performance-benchmarking-mysql-57-vs-mysql-80

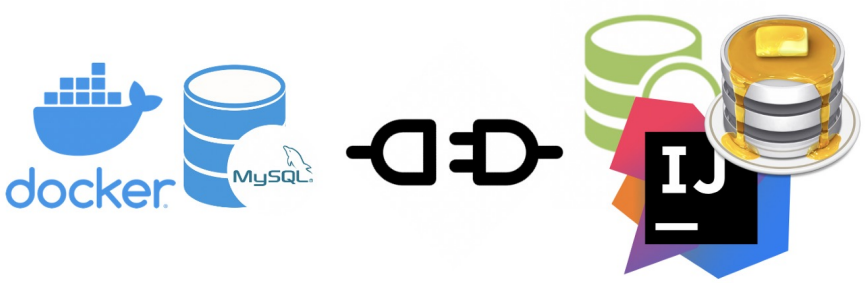
Where to Get MySQL?

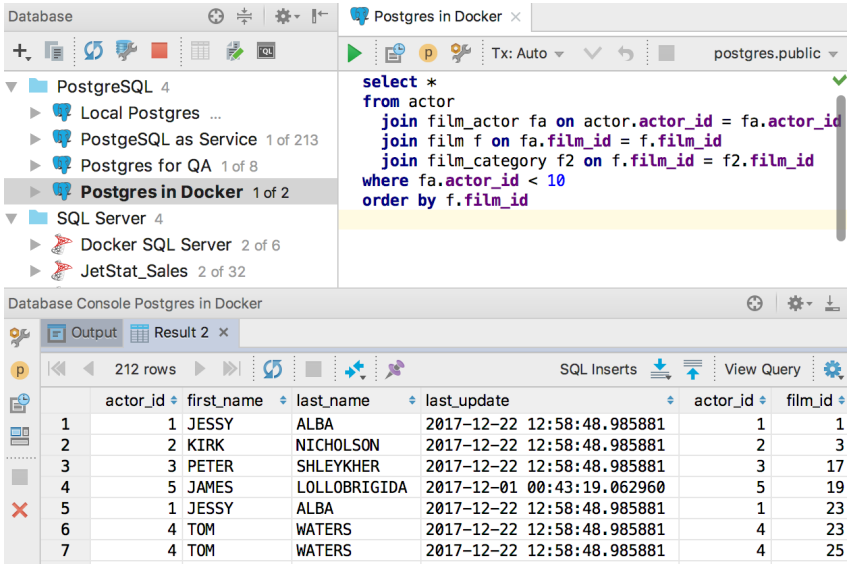
- https://dev.mysql.com/downloads/
  - Look for the “Community” versions – the branch that is available for free
  - ”Enterprise” versions are the commercial ones

- We have prepared a Docker image for the course
  - Consists of Ubuntu Server, MySQL, example databases for course activities

* Image src: https://www.docker.com

Where to Get MySQL?





	actor_id	first_name	last_name	last_update	actor_id	film_id
1	1	JESSY	ALBA	2017-12-22 12:58:48.985881	1	1
2	2	KIRK	NICHOLSON	2017-12-22 12:58:48.985881	2	3
3	3	PETER	SHLEYKHER	2017-12-22 12:58:48.985881	3	17
4	5	JAMES	LOLLOBRIGIDA	2017-12-01 00:43:19.062960	5	19
5	1	JESSY	ALBA	2017-12-22 12:58:48.985881	1	23
6	4	TOM	WATERS	2017-12-22 12:58:48.985881	4	23
7	4	TOM	WATERS	2017-12-22 12:58:48.985881	4	25

* Image source: https://baumannalexj.medium.com/connect-your-db-tool-to-a-dockerized-mysql-server-container-bc18853524ed
https://www.jetbrains.com/datagrip/features/look_and_feel.html

Agenda

- Introduction to MySQL
- **SQL preview**

Structured Query Language (SQL)

- SQL: Structured Query Language
  - The principal language used to describe and manipulate relational databases
  - Very high-level
    - Say “what to do” rather than “how to do it”
    - SQL is not specifying data-manipulation details
    - DBMSs figure out the “best” way to execute queries
      - Called “query optimization”
  - Two aspects to SQL
    - Data definition: for declaring database schemas (DDL)
    - Data manipulation: for querying (asking questions about) databases and for modifying the database (DML)

SQL Parts

- DML – provides the ability to query information from the database and to insert tuples into, delete tuples from, and modify tuples in the database
- Integrity – the DDL includes commands for specifying integrity constraints
- Vide definition – the DDL includes commands for defining views

- Transaction control – includes commands for specifying the beginning and ending of transactions
- Embedded SQL and dynamic SQL – define how SQL statements can be embedded within general-purpose programming language
- Authorization – includes commands for specifying access rights to relations and views

A Brief History

- IBM SEQUEL (Structured English Query Language) was developed as a part of the System R project (Chamberlin and Boyce, early 1970s)
  - Later on, SEQUEL was renamed SQL (structured query language)
  - System R → System/38 (1979), SQL/DS (1981), DB2 (1983)
- Relational Software, Inc released the first commercial implementation of SQL, Oracle V2 for VAX computers
  - Relational Software, Inc is now Oracle Corporation
- ANSI and ISO standardized SQL:
  - SQL-86, SQL-89, SQL-92, SQL:1999, …, SQL:2011, SQL:2016 (current)
  - SQL-92 is supported by the most of database systems

Basic Query Structure

- A typical SQL query has the form:

  SELECT $A_1, A_2, ..., A_n$
  FROM $r_1, r_2, ..., r_m$
  WHERE P

  - A_i represents an attribute
  - R_i represents a relation
  - P is a predicate

- The result of an SQL query is a relation

EOF

- Coming next:
  - Structured Query Language