CS 764: Topics in Database Management Systems

Lecture 1: Introduction

Xiangyao Yu

09/04/2025

Who am I?

Name: Xiangyao Yu

Assistant professor in Computer Sciences, Database Group

Research interests:

- GPU databases
- Cloud-native databases
- Core DB techniques

Basic Information

Course website: https://pages.cs.wisc.edu/~yxy/cs764-f25/index.html

Instructor: Xiangyao Yu

Office hours: schedule by email

TA: Devesh Sarda

Office hours: schedule by email

Today’s Agenda

Database 101

Course logistics

Database 101

Database: A collection of data, typically describing the activities of one or more related organizations. For example:

- Entities: students, instructors, courses
- Relationships: students enroll in courses, instructors teach courses

Database 101

Database: A collection of data, typically describing the activities of one or more related organizations. For example:

- Entities: students, instructors, courses
- Relationships: students enroll in courses, instructors teach courses

Database management system (DBMS): Software designed to assist in **maintaining** and **utilizing large collection** of data.

Relational Model

A relational database is a **collection of one or more relations**, where each relation is a **table with rows and columns**.

An example relation:

table name

Product

name	category	price	manufacturer
iPad	tablet	$399.00	Apple
Surface	tablet	$299.00	Microsoft
…	…	…	…

Relational Model

A relational database is a **collection of one or more relations**, where each relation is a **table with rows and columns**.

An example relation:

table name

Product

name	category	price	manufacturer
iPad	tablet	$399.00	Apple
Surface	tablet	$299.00	Microsoft
…	…	…	…

record/tuple/row

Relational Model

A relational database is a **collection of one or more relations**, where each relation is a **table with rows and columns**.

An example relation:

table name

Product

attribute/column

name	category	price	manufacturer
iPad	tablet	$399.00	Apple
Surface	tablet	$299.00	Microsoft
…	…	…	…

record/tuple/row

SQL Queries

SELECT $a_1, a_2, \ldots, a_k$

FROM $R_1, R_2, \ldots, R_n$

WHERE conditions

A Database Template

SELECT $a_1, a_2, \ldots, a_k$

FROM $R_1, R_2, \ldots, R_n$

WHERE conditions

Vanilla query executor

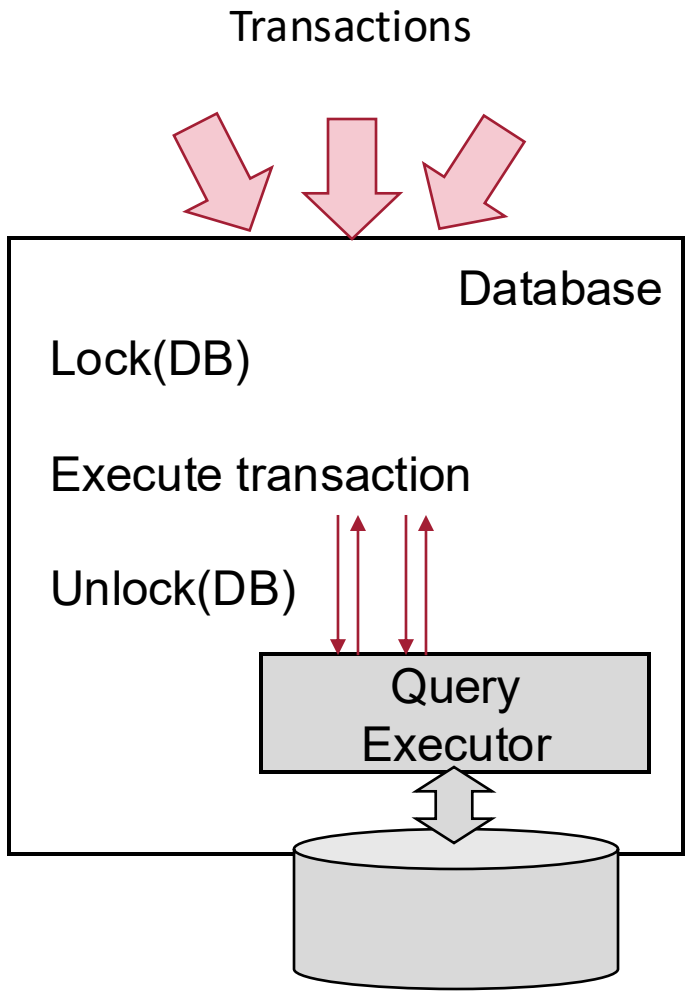
```
answer = {}
for t1 in R1 do
  for t2 in R2 do
    …
    for tn in Rn do
      if conditions
        then answer = answer U {(a1,…,ak)}
return answer
```

A Database Template

SELECT $a_1, a_2, \ldots, a_k$
FROM $R_1, R_2, \ldots, R_n$
WHERE conditions

Vanilla query executor

```
answer = {}
for t1 in R1 do
  for t2 in R2 do
    …
    for tn in Rn do
      if conditions
        then answer = answer U {(a1,…,ak)}
return answer
```

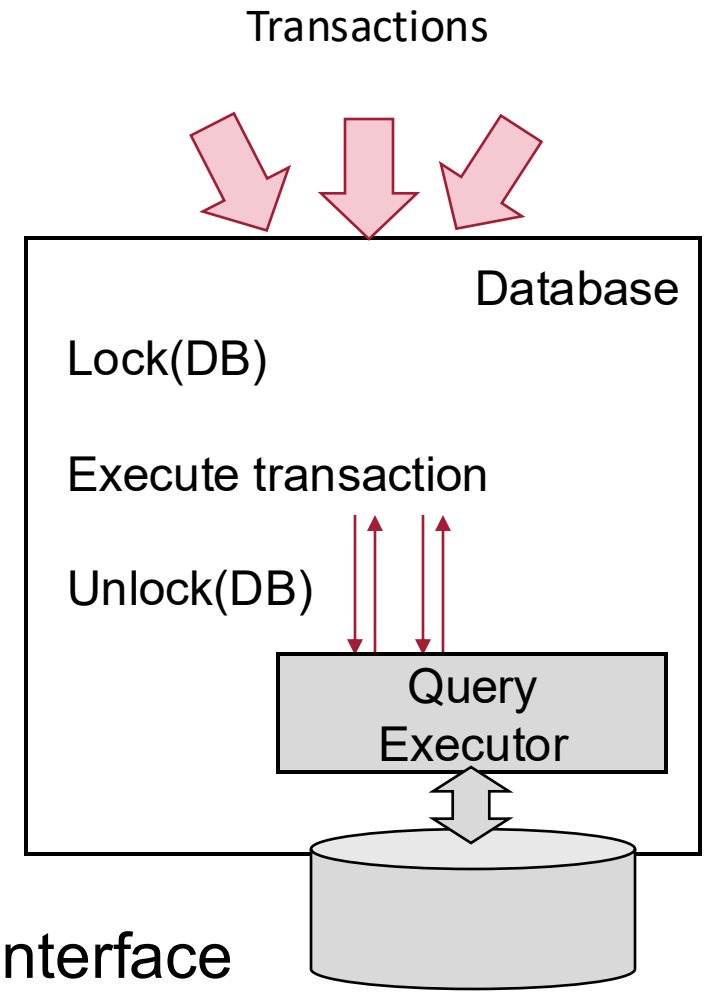


A Database Template

SELECT $a_1, a_2, \ldots, a_k$
FROM $R_1, R_2, \ldots, R_n$
WHERE conditions

Vanilla query executor

```
answer = {}
for t1 in R1 do
  for t2 in R2 do
    …
    for tn in Rn do
      if conditions
        then answer = answer U {(a1,…,ak)}
return answer
```



A DBMS can be heavily optimized beneath this simple interface

Optimizing the Template Implementation

SELECT $a_1, a_2, \ldots, a_k$
FROM $R_1, R_2, \ldots, R_n$
WHERE conditions

Vanilla query executor

```
answer = {}
for t1 in R1 do
  for t2 in R2 do
    …
      for tn in Rn do
        if conditions
          then answer = answer ∪ {(a1,…,ak)}
return answer
```

Cross products are expensive, can replace with **joins**

Optimizing the Template Implementation

SELECT $a_1, a_2, \ldots, a_k$
FROM $R_1, R_2, \ldots, R_n$
WHERE conditions

Vanilla query executor

```
answer = {}
for t1 in R1 do
  for t2 in R2 do
    …
      for tn in Rn do
        if conditions
          then answer = answer ∪ {(a1,…,ak)}
return answer
```

Cross products are expensive, can replace with **joins**

Avoid scanning the entire table by accessing subsets of records through an **index**

Optimizing the Template Implementation

SELECT $a_1, a_2, \ldots, a_k$
FROM $R_1, R_2, \ldots, R_n$
WHERE conditions

Vanilla query executor

```
answer = {}
for t1 in R1 do
  for t2 in R2 do
    …
      for tn in Rn do
        if conditions
          then answer = answer ∪ {(a1,…,ak)}
return answer
```

Cross products are expensive, can replace with **joins**

Avoid scanning the entire table by accessing subsets of records through an **index**

Query plan can be **optimized** to minimize the execution overhead

Optimizing the Template Implementation

SELECT $a_1, a_2, \ldots, a_k$
FROM $R_1, R_2, \ldots, R_n$
WHERE conditions

Data can be stored in disks for persistency and low cost and **buffered** in DRAM

Vanilla query executor

```
answer = {}
for t1 in R1 do
  for t2 in R2 do
    …
      for tn in Rn do
        if conditions
          then answer = answer ∪ {(a1,…,ak)}
return answer
```

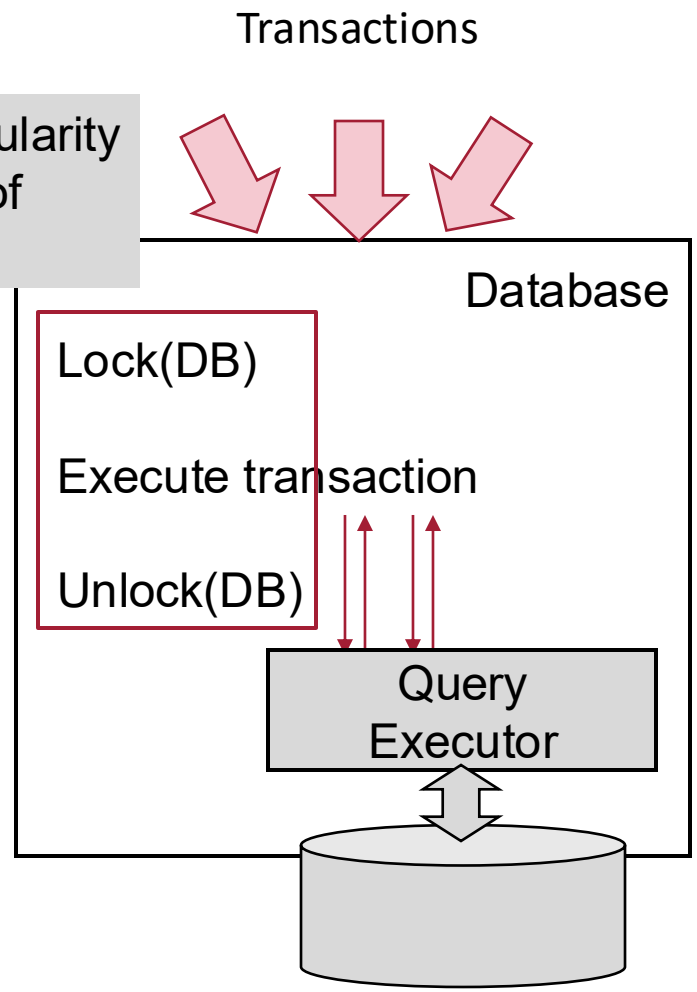
Cross products are expensive, can replace with **joins**

Avoid scanning the entire table by accessing subsets of records through an **index**

Query plan can be **optimized** to minimize the execution overhead

Optimizing the Template Implementation

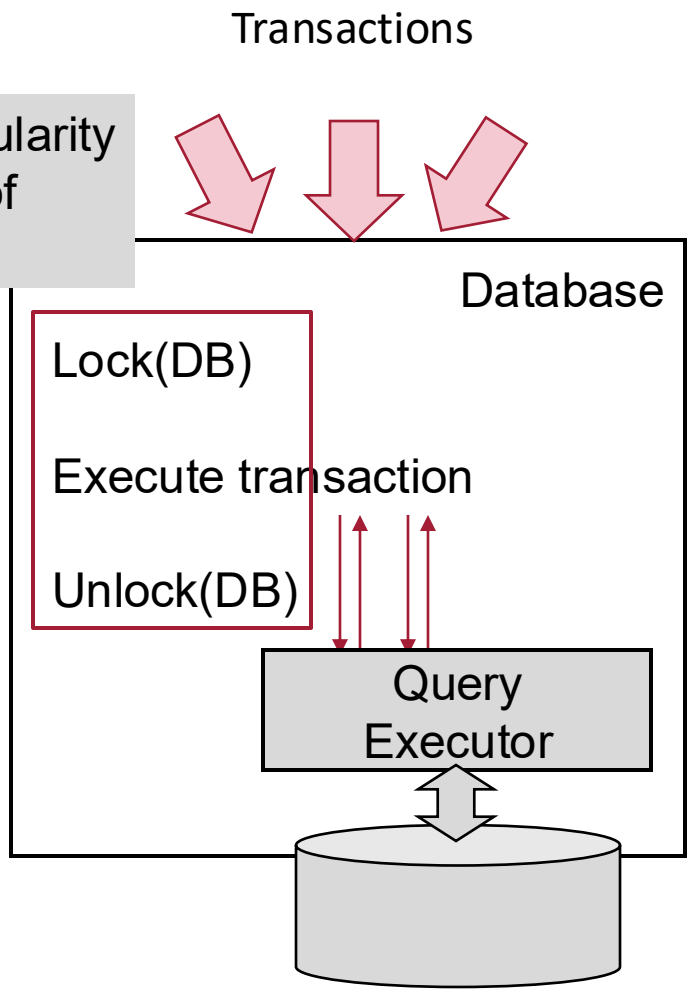
Perform **locking** at fine-granularity to enable parallel execution of transactions



Optimizing the Template Implementation

Perform **locking** at fine-granularity to enable parallel execution of transactions

Ensure that parallel execution results are **equivalent to serial** execution

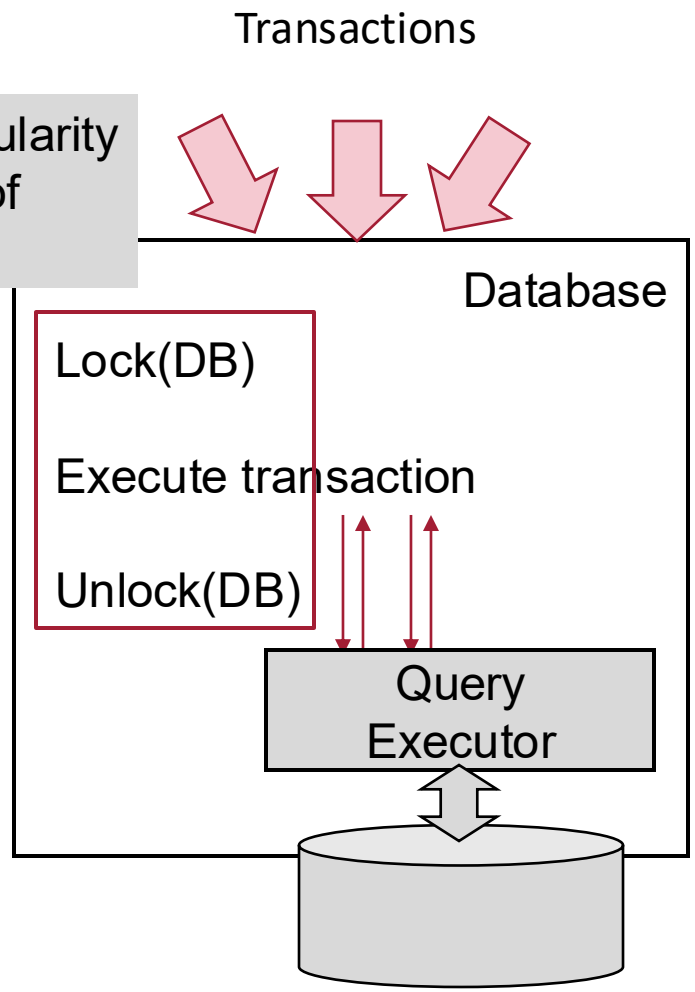


Optimizing the Template Implementation

Perform **locking** at fine-granularity to enable parallel execution of transactions

Ensure that parallel execution results are **equivalent to serial** execution

Ensure the database can tolerate failures by providing **durability and high availability**



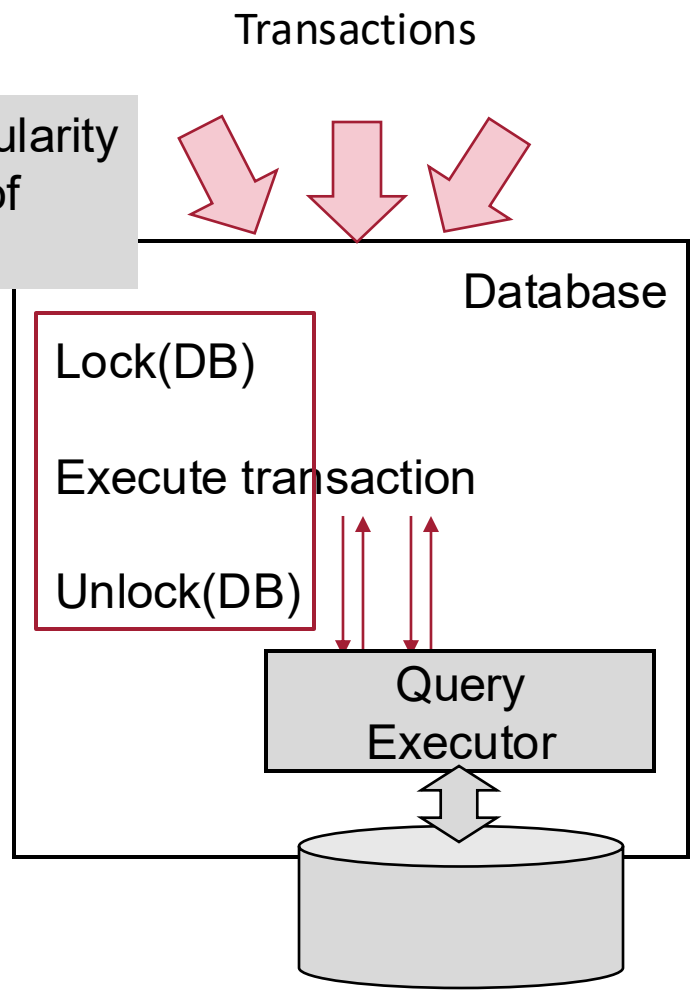
Optimizing the Template Implementation

Perform **locking** at fine-granularity to enable parallel execution of transactions

Ensure that parallel execution results are **equivalent to serial** execution

Ensure the database can tolerate failures by providing **durability and high availability**

Can **scale up** to multicore processors and **scale out** to distributed systems



Topics in CS 764

Analytical query processing (~10 Lectures)

- Join
- Buffer management
- Query optimization
- Column store
- Parallel database
- GPU databases

Transaction processing (~10 Lectures)

- Two-phase locking
- Isolation
- Optimistic concurrency control
- B-tree and radix-tree
- Fault tolerance

Guest lecture from Snowflake

Course Logistics

Course Information

Course Website: http://pages.cs.wisc.edu/~yxy/cs764-f25/

Canvas: https://canvas.wisc.edu/courses/464260

Piazza: https://piazza.com/class/mf4idsrcroiqx (can be accessed through Canvas as well)

Prerequisite: CS 564

Reference textbooks:

- Red book
- Cow book

Grading

Paper review: 15%

Exam: 35%

Project proposal: 10%

Project presentation: 10%

Project final report: 30%

Paper Review (15%)

Paper reading: one classic/modern paper per lecture

- username: cs764 password: dbguru

Paper Review (15%)

Paper reading: one classic/modern paper per lecture

- username: cs764 password: dbguru

Upload review: https://wisc-cs764-f25.hotcrp.com (must submit before the lecture starts in order to be graded)

- Overall merit
- Paper summary
  - What main research problem/challenge did the paper address?
  - What is the key contribution of the paper?
- **Comments and questions**
  - Aspects you like or dislike about the paper
  - Questions about that paper that you wish to be discussed in the lecture

Paper Review (15%)

Paper reading: one classic/modern paper per lecture

- username: cs764 password: dbguru

Upload review: https://wisc-cs764-f25.hotcrp.com (must submit before the lecture starts in order to be graded)

- Overall merit
- Paper summary
  - What main research problem/challenge did the paper address?
  - What is the key contribution of the paper?
- **Comments and questions**
  - Aspects you like or dislike about the paper
  - Questions about that paper that you wish to be discussed in the lecture

Grading: You can skip up to 2 reviews without losing points; otherwise 1% of total grade (up to 15%) is deducted for each missing review

Exam (35%)

Take-home exam

- Open-book, open-notes
- You can use any material provided in this course or on the Internet

Sample exam questions are available on course website

Course Project (50%)

In **groups of 2–4** students

Project ideas will be provided but you are encouraged to propose your own ideas

- Project ideas for Fall 2020–2022 are available on the course website
- Three example projects are available on the course website (two papers based on course projects accepted to SIGMOD 2022 and SIGMOD 2023)

Course Project (50%)

In **groups of 2–4** students

Project ideas will be provided but you are encouraged to propose your own ideas

- Project ideas for Fall 2020–2022 are available on the course website
- Three example projects are available on the course website (two papers based on course projects accepted to SIGMOD 2022 and SIGMOD 2023)

Important dates

- Create teams and submit proposal: Oct. 17
- Project meetings with instructor: TBD
- Presentation: Dec. 2 & 4
- Paper submission: Dec. 15

Computation Resources

CloudLab

https://www.cloudlab.us/signup.php?pid=NextGenDB (project name: NextGenDB)

Chameleon

https://www.chameleoncloud.org (project name: ngdb)

Waitlist

Class size limited to ~65

If you are enrolled but don’t want to take the class, please drop ASAP

If you are on the waitlist, we will admit students first-come-first-serve

Before next lecture

Read the following paper and submit review

- Leonard D. Shapiro, Join Processing in Database Systems with Large Main Memories. ACM Trans. Database Syst. 1986.

Email the instructor if you have problems registering for https://wisc-cs764-f25.hotcrp.com after Friday