COMP90050
Advanced Database Systems

THE UNIVERSITY OF MELBOURNE

Winter Semester, 2023

Lecturer: Farhana Choudhury (PhD)

Live lecture – week 2

Quizzes

Quiz 1 and Quiz 2 are done!

Time to discuss the solutions

Project

Step 1: Form a group of **4 students** by 2 July, 2023, 11:59pm

Step 2: Pick a topic of your interest from a list of candidate topics provided, by 2 July, 2023, 11:59pm

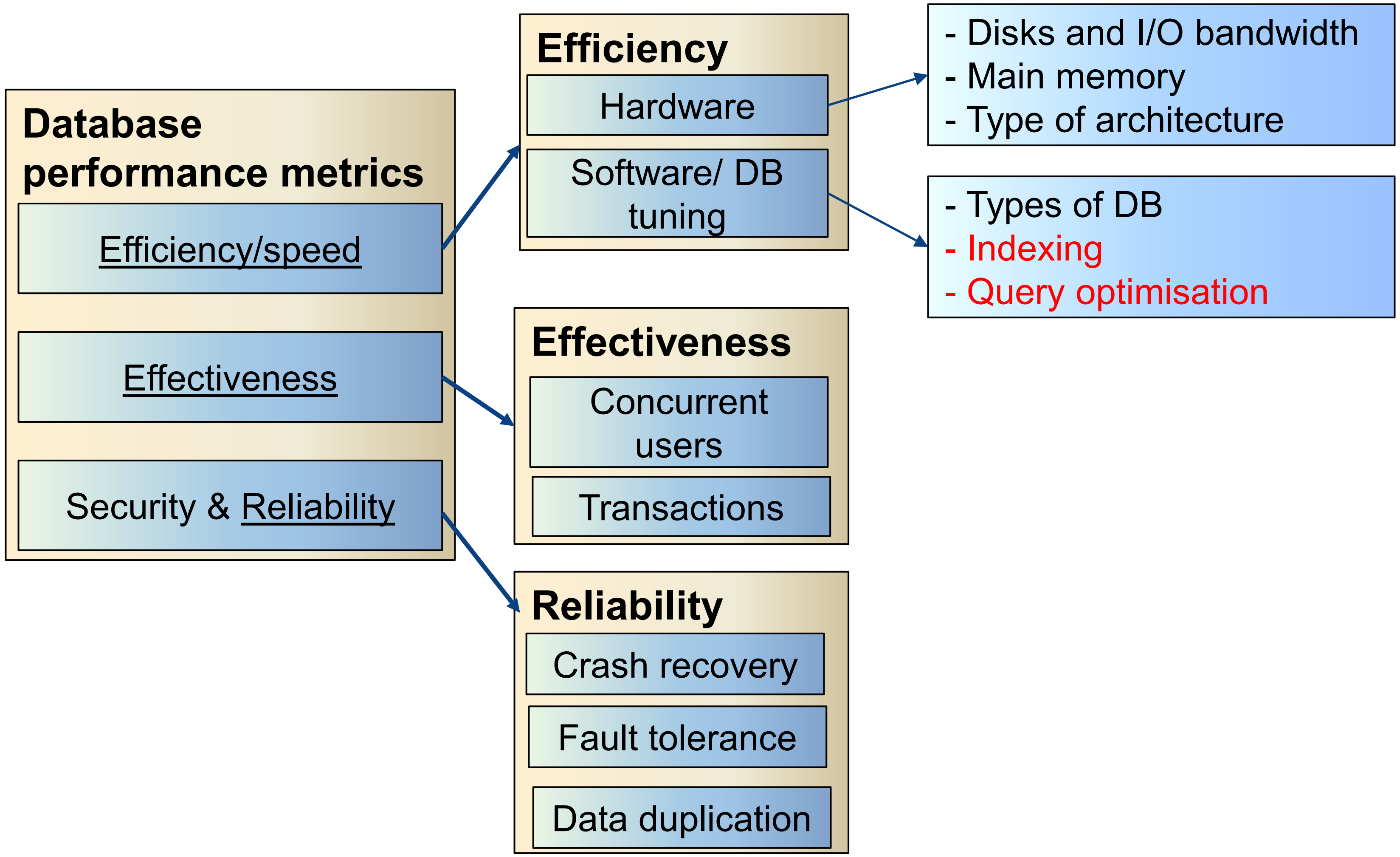
Canvas has details about the projects already: **Read** and **Start** with the steps above.

Presentation: During Week 4 (schedule will be uploaded)

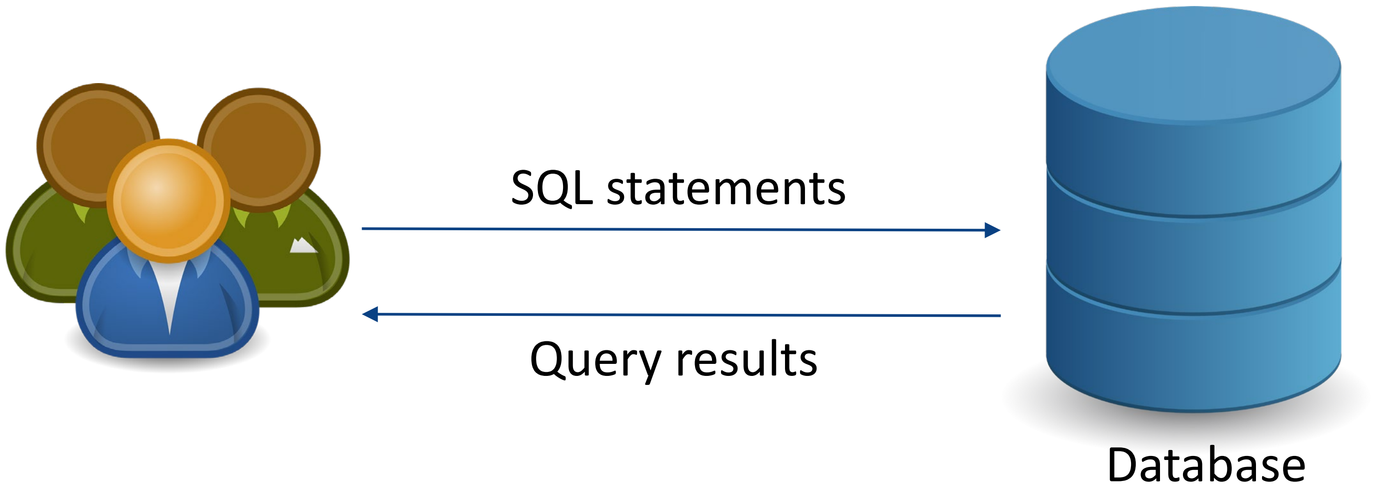
Report submission due data: 24 July 2023, 11:59pm

Discussion on the topics of the pre-recorded lecture and additional contents

Core Concepts of Database management system



DB queries



Efficiency is one of the most important requirements for client-facing databases

The best execution strategy to find the query results – an integral part of DBMS

How do query optimizer make the choices?

Steps in **cost-based query optimization**

1. Generate logically equivalent expressions of the SQL statement

2. Annotate resultant expressions to get alternative query plans

3. Choose the cheapest plan based on the estimated cost

Estimation of plan cost based on:

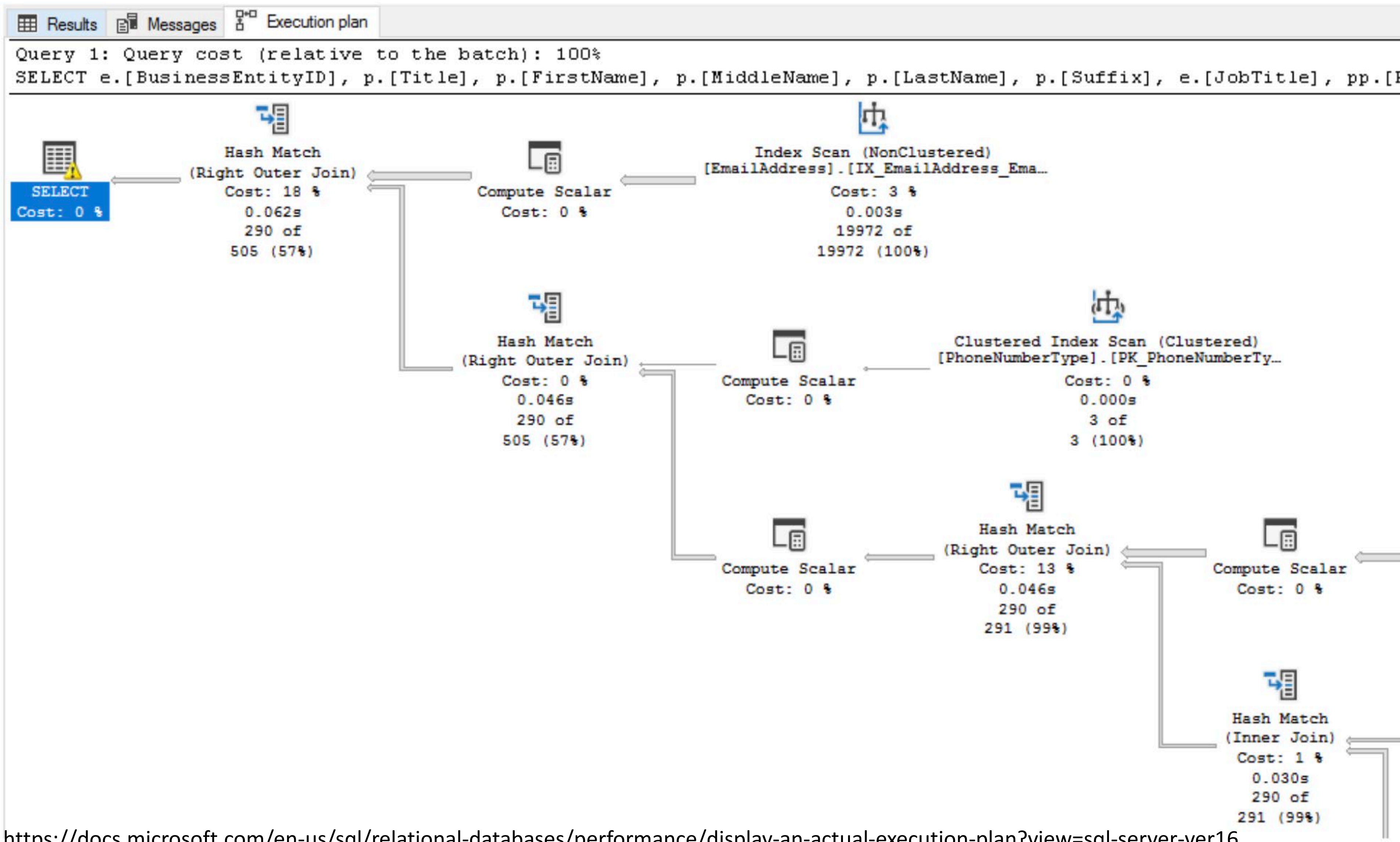
- Statistical information about tables. Example:

  number of distinct values for an attribute

- Statistics estimation for intermediate results to compute cost of complex expressions

- Cost formulae for algorithms, computed using statistics again

View an actual execution plan



```
Query 1: Query cost (relative to the batch): 100%
SELECT e.[BusinessEntityID], p.[Title], p.[FirstName], p.[MiddleName], p.[LastName], p.[Suffix], e.[JobTitle], pp.[P
```

https://docs.microsoft.com/en-us/sql/relational-databases/performance/display-an-actual-execution-plan?view=sql-server-ver16

Query optimisation

The goal is to minimise the number of disk blocks (e.g., pages) reads
- Is the query optimiser optimistic or pessimistic about the query cost estimation?

Time for a poll - Pollev.com/farhanachoud585

Performance tuning

When you identify a query with suboptimal performance, what can you do?

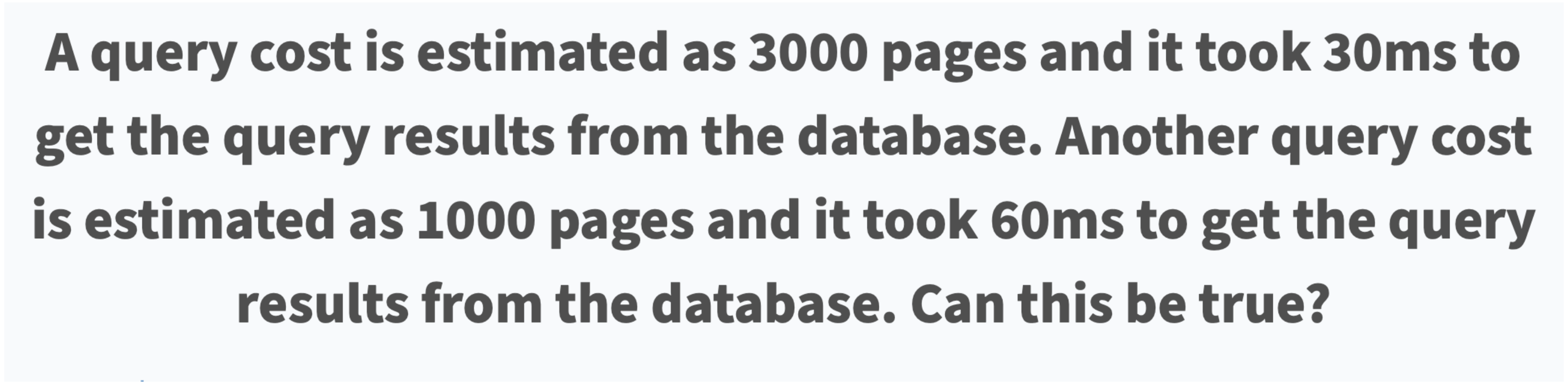
Time for another poll - Pollev.com/farhanachoud585

- Force a query plan instead of the plan chosen by the optimizer
- Do we need an index?
- Enforce statistic recompilation
- Rewrite query?

Source: https://docs.microsoft.com/en-us/sql/relational-databases/performance/best-practice-with-the-query-store?view=sql-server-ver16#Insight

Query optimisation

The goal is to minimise the number of disk blocks (e.g., pages) reads
- Is the query optimiser optimistic or pessimistic about the query cost estimation?


A query cost is estimated as 3000 pages and it took 30ms to get the query results from the database. Another query cost is estimated as 1000 pages and it took 60ms to get the query results from the database. Can this be true?

What if some pages are already in main memory? –
Difference between logical reads and physical reads

Memory optimised table

Can you make a table stay in main memory?

```
CREATE TABLE dbo.Customer (
    CustomerID char (5) NOT NULL PRIMARY KEY,
    ContactName varchar (30) NOT NULL
) WITH (MEMORY_OPTIMIZED=ON)
```

Query processing and query optimisation for memory optimised tables?

12

Indexing is Critical for Efficiency

- An **index file** consists of records (called **index entries**) of the form <u>**search-key, pointer to where data is**</u>
- Index files are typically much smaller than the original data files and many parts of it are already in memory

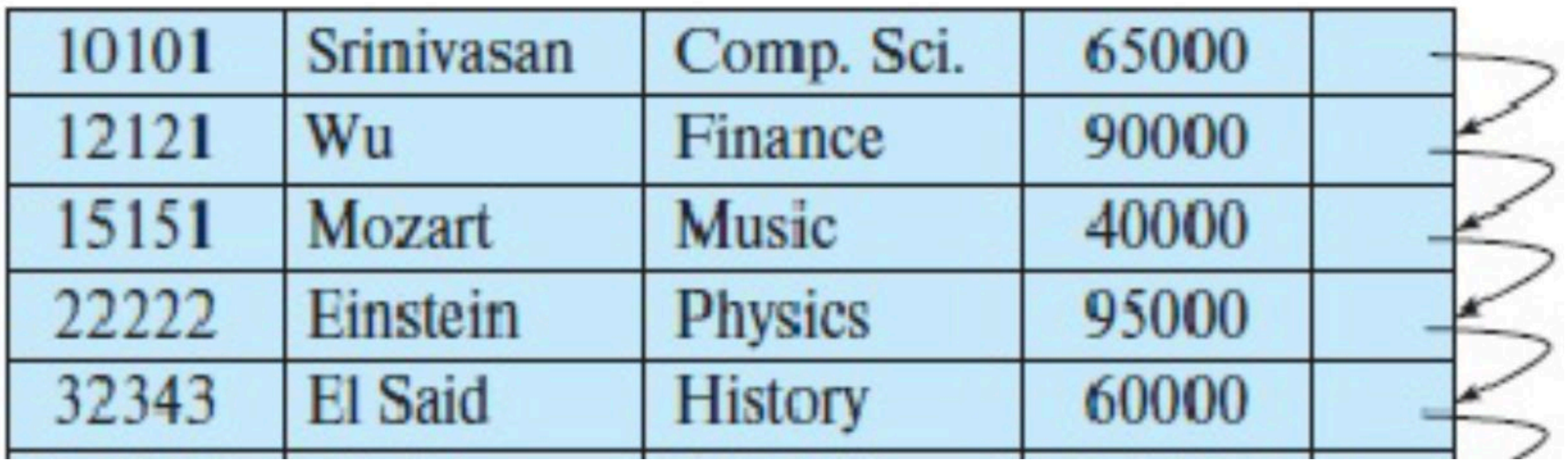
Indexing is Critical for Efficiency

We have seen some example index structures

- B+ tree
- Hash index
- Bitmap index
- Spatial indexes
    - Quadtree
    - R-tree

Effect of indexes – do they always improve performance?

Time for a poll - Pollev.com/farhanachoud585



10101	Srinivasan	Comp. Sci.	65000	
12121	Wu	Finance	90000	
15151	Mozart	Music	40000	
22222	Einstein	Physics	95000	
32343	El Said	History	60000	

Query: Find the names of the instructors teaching ‘Comp. Sci.’

Scenario 1: No indexes

Scenario 2: B+tree index on IDs

Scenario 3: Both of the above will have the same performance

Effect of indexes – do they always improve performance?

Time for a poll - Pollev.com/farhanachoud585

ID	Name	Department
1	Jane	Comp. Sci
2	John	Biology

ID	Position	Salary
1	Lecturer	75,000
1	Research assistant	40,000
2	Senior lecturer	82,000

Query: Find the names of the instructors who are ‘lecturer’

- Scenario 1: No indexes
- Scenario 2: B+tree index on IDs
- Both of the above will have the same performance
- Depends on statistics (e.g., rows that need to be joined)

Effect of indexes – do they always improve performance?

ID	Name	Department
1	Jane	Comp. Sci
2	John	Biology

ID	Position	Salary
1	Lecturer	75,000
1	Research assistant	40,000
2	Senior lecturer	82,000

Query: Find the names of the instructors with salary>80000

>0

Scenario 1: No indexes

Scenario 2: B+tree index on IDs

Both of the above will have the same performance

It will depend on statistics (e.g., rows that need to be joined)

Effect of indexes – do they always improve performance?

Breakout room activity - https://tinyurl.com/5czrbhff

ID	Name	Department
1	Jane	Comp. Sci
2	John	Biology

ID	Position	Salary
1	Lecturer	75,000
1	Research assistant	40,000
2	Senior lecturer	82,000

There are indexes constructed on:

- B+tree on IDs of both tables
- Hash index on Department
- Bitmap index on position

What will be the performance of data insertion, deletion, and updates, compared to no indexing (or just on IDs)

Search using Indexes

We have seen some example index structures

- B+ tree – finding a particular value, finding a range of values
- Hash index – finding a particular value
- Bitmap index – finding total number
- Spatial indexes – range query, nearest neighbor query
  - Quadtree
  - R-tree

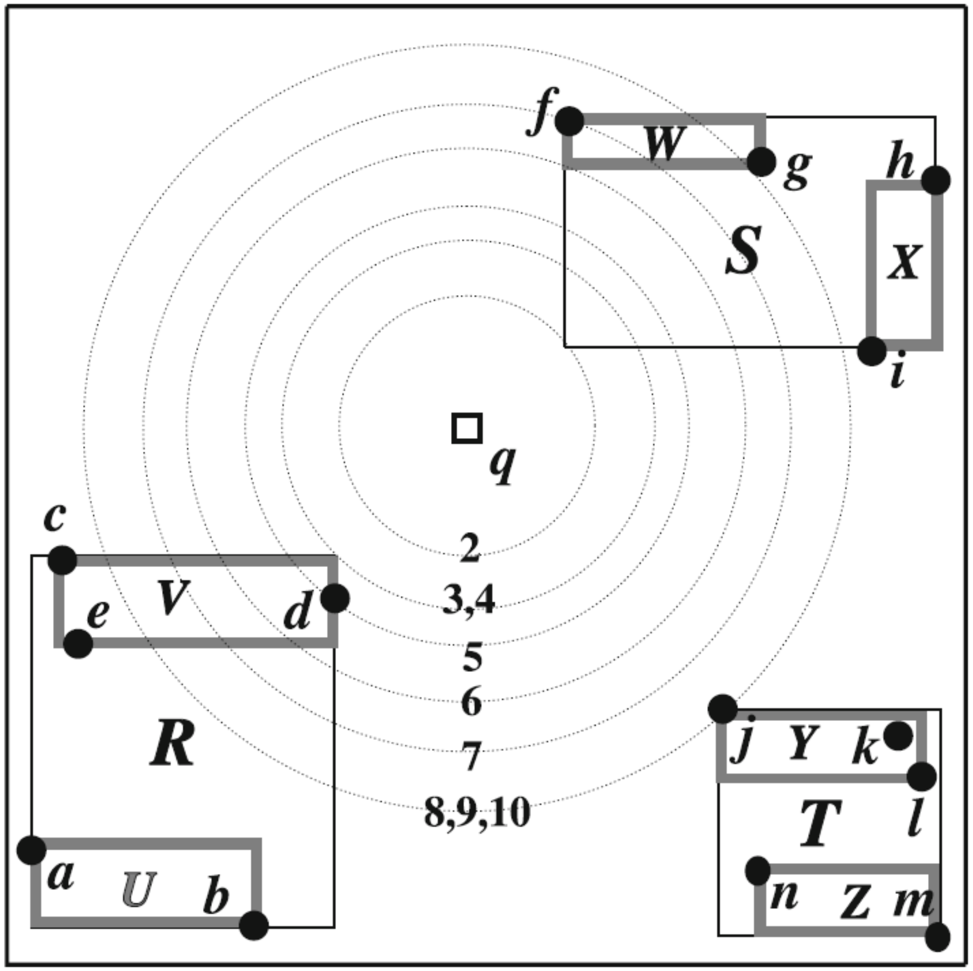
Nearest neighbor in R-Trees

To find the nearest neighbor of a given query point/region, do the following, starting from the root node:

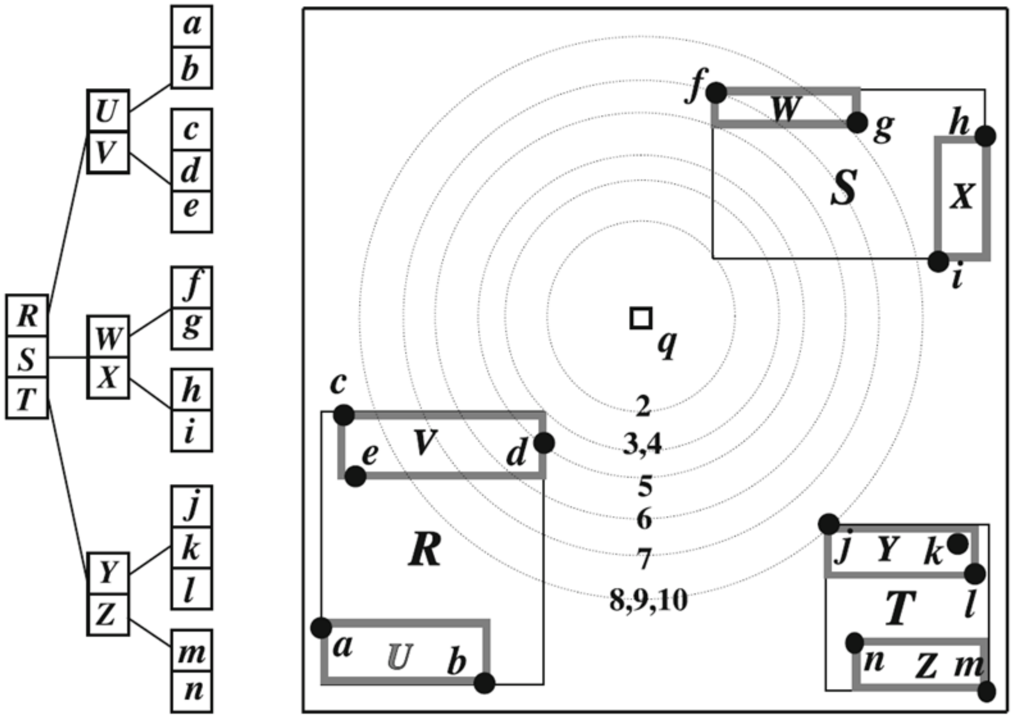
- Use a sorted priority queue of the R-tree nodes based on the minimum distance from the query
- Traverse the node that is in the top of the priority queue, and put its elements in the queue. Continue
- Stop when the top node is a data object (first NN has been found)

This algorithm is a best-first search algorithm

Example R-tree



Nearest Neighbor Query on R-tree



Step	Priority queue	Retrieved NN(s)
1	$\langle S, R, T\rangle$	$\langle\rangle$
2	$\langle R, W, T, X\rangle$	$\langle\rangle$
3	$\langle V, W, T, X, U\rangle$	$\langle\rangle$
4	$\langle \mathbf{d}, W, T, X, \mathbf{c}, \mathbf{e}, U\rangle$	$\langle\rangle$

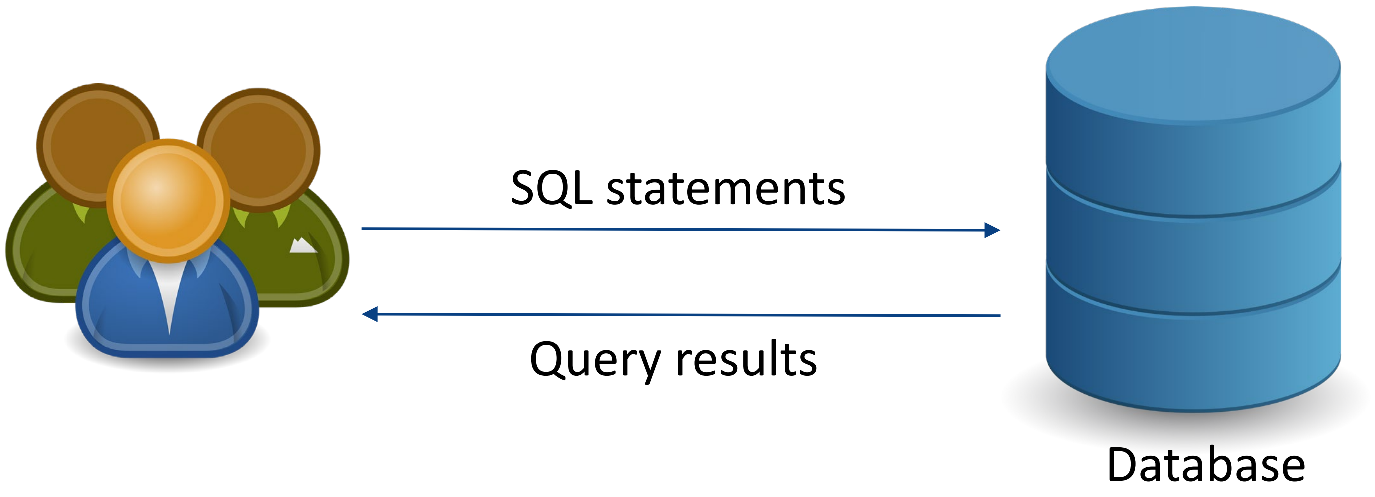
Why do we look at the search / query algorithms?

Step 5 finds d as the first NN (using Best First search)

What you need to do now?

- Given a data set, when uploading to the DBMS
  - Find the potential query types
  - Research what indices that particular DBMS would have for that data type
  - Research for what queries you would better do on what index
  - Create index if you have large data
  - Monitor performance
  - Tune or create other indices

Discussion topic – SQL injection



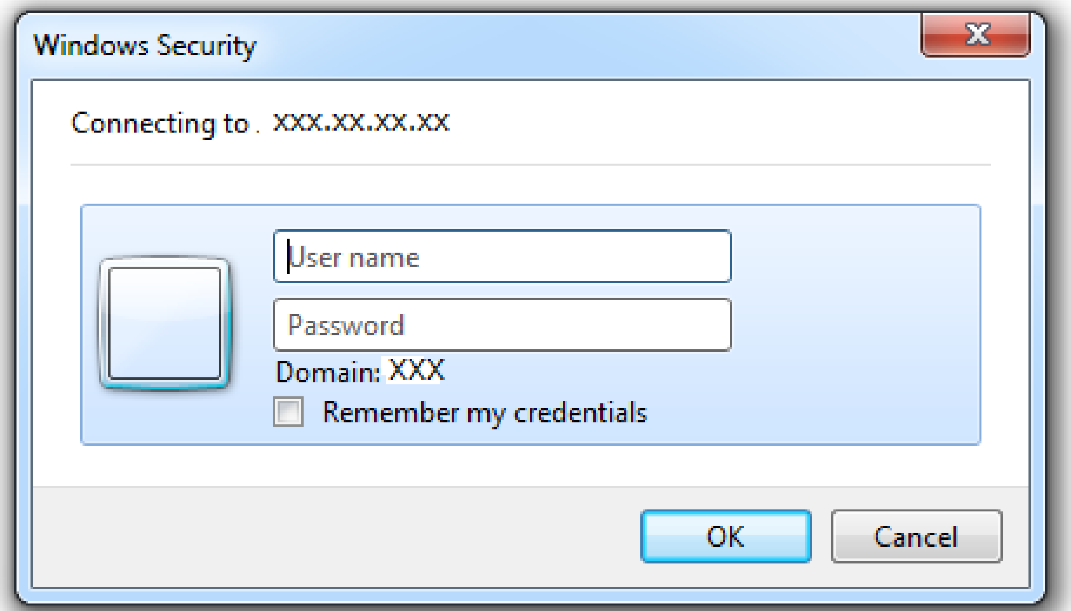
Can happen when an application executes database query using user-input data, and the user input or part of the user input is treated as SQL statement

SQL syntax

LOGIC: ‘a’=‘a’

Example: SELECT * FROM `table` WHERE ‘a’=‘a’

SELECT * FROM `login` WHERE `user`=‘farhana’ AND `pass`=‘comp90050’



Windows Security

Connecting to . XXX.XX.XX.XX

Domain: XXX

Remember my credentials

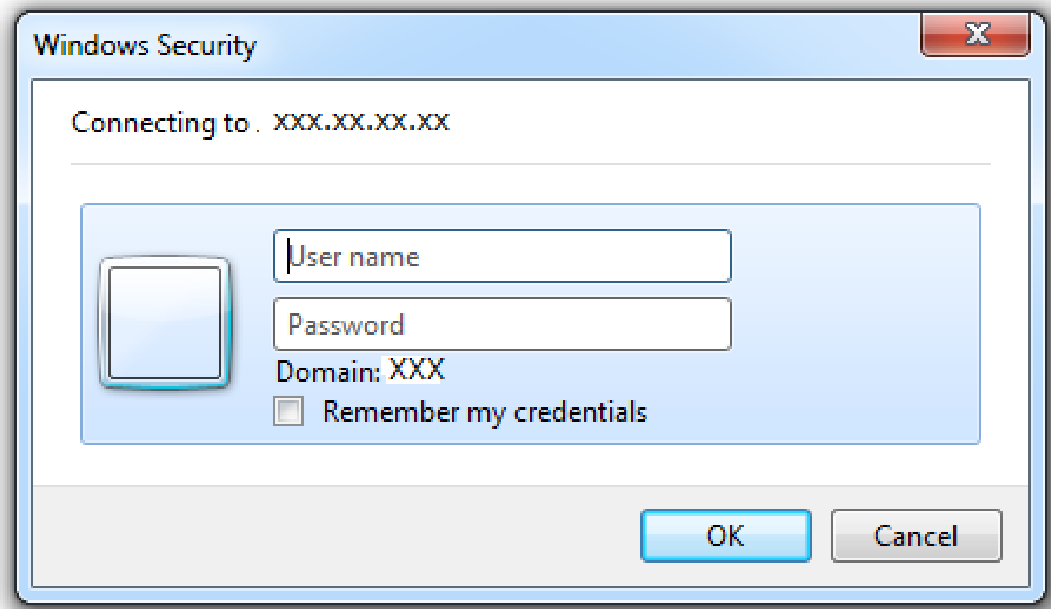
SELECT * FROM `login` WHERE `user`=‘’ OR ‘a’=‘a’ AND `pass`=‘’ OR ‘a’=‘a’

SQL syntax

MULTI STATEMENTS: S1; S2

Example: SELECT * FROM `table`; DROP TABLE `table`;

SELECT * FROM `login` WHERE `user`='farhana' AND `pass`='comp90050'


Windows Security

Connecting to . XXX.XX.XX.XX

Domain: XXX

☐ Remember my credentials

Any statement(s) can go here

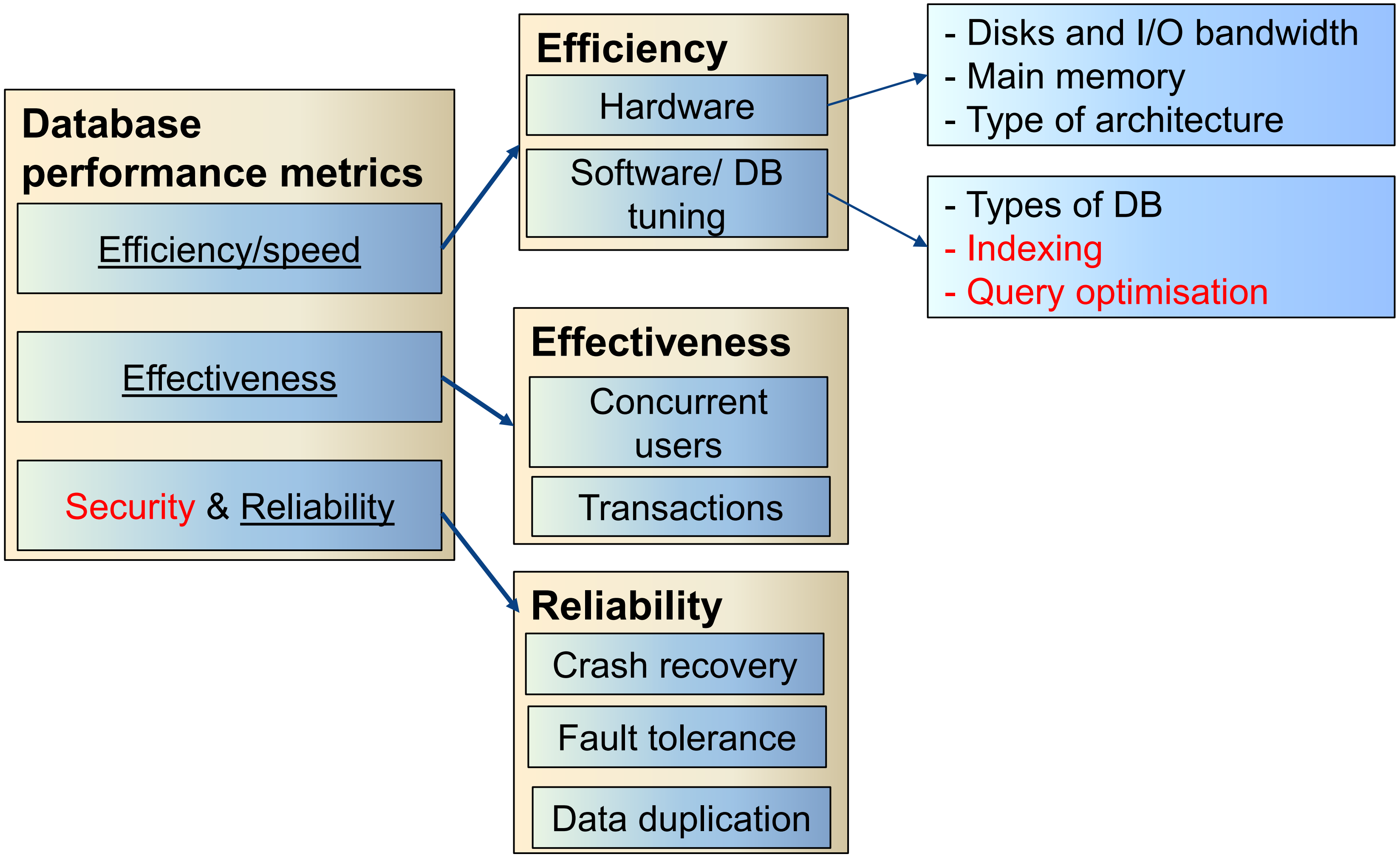
SELECT * FROM `login` WHERE `user`=''; DROP TABLE `login`; --' AND `pass`=''

Prevention

User parameterized query/prepared statement - allows the database to distinguish between code and data

```
String query = "SELECT * from login where user = "
+ request.getParameter(”userName");
```

Summary


Database performance metrics
Efficiency/speed
Effectiveness
Security & Reliability
Efficiency
Hardware
Software/ DB tuning
Effectiveness
Concurrent users
Transactions
Reliability
Crash recovery
Fault tolerance
Data duplication
- Disks and I/O bandwidth
- Main memory
- Type of architecture
- Types of DB
- Indexing
- Query optimisation