Module 4

Optmizing SQL Code

Created by Teerachai Laothong

Agenda

- Using Bind Variables
- Avoiding Full Table Scans
- Database Index
- Using Index

Using Bind Variables

- Avoiding dynamic SQL
- Latch
- "Bad" application might slow down all systems

Bind Variables

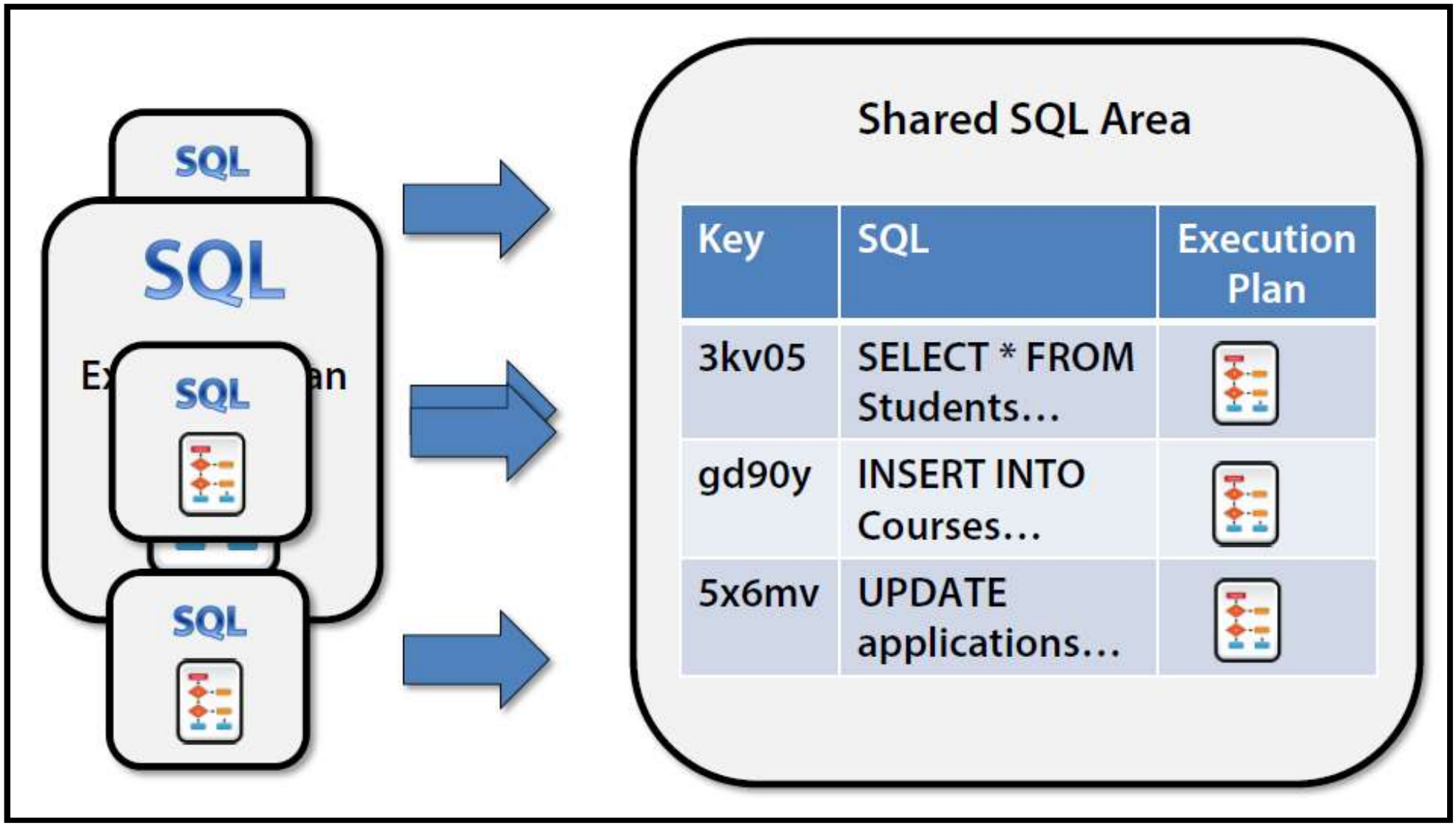
A bind variable is a placeholder, used in our SQL statement, which can bind with actual values during execution.

```
SELECT * FROM sh.customers s WHERE s.cust_id =:p_cust_id
```

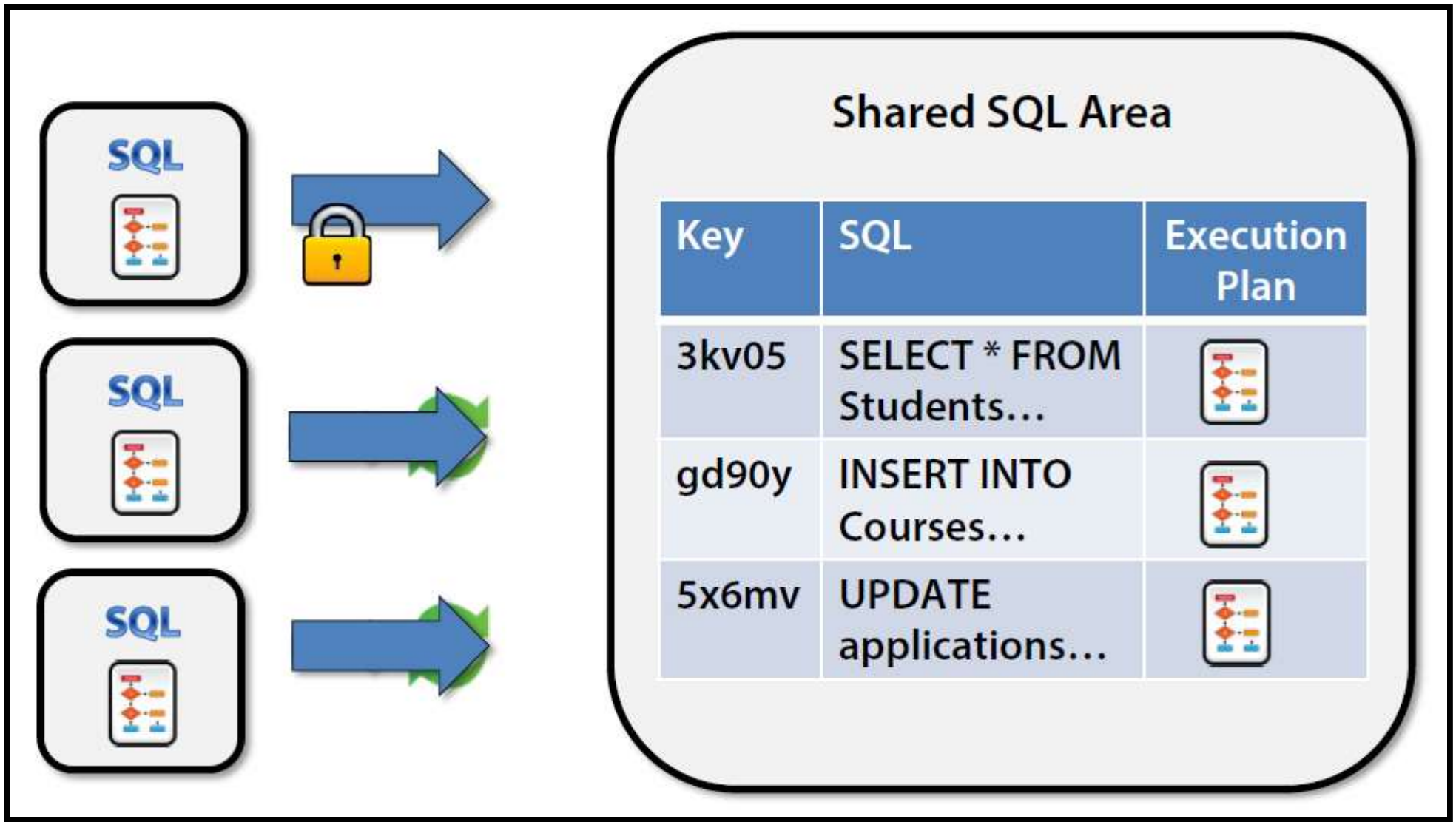
Bind Variables and Parsing

When using bind variables, the query is being **hard parsed** for the first time,
and subsequent calls to the same statements—but with different parameters—will require only a **soft parse**.

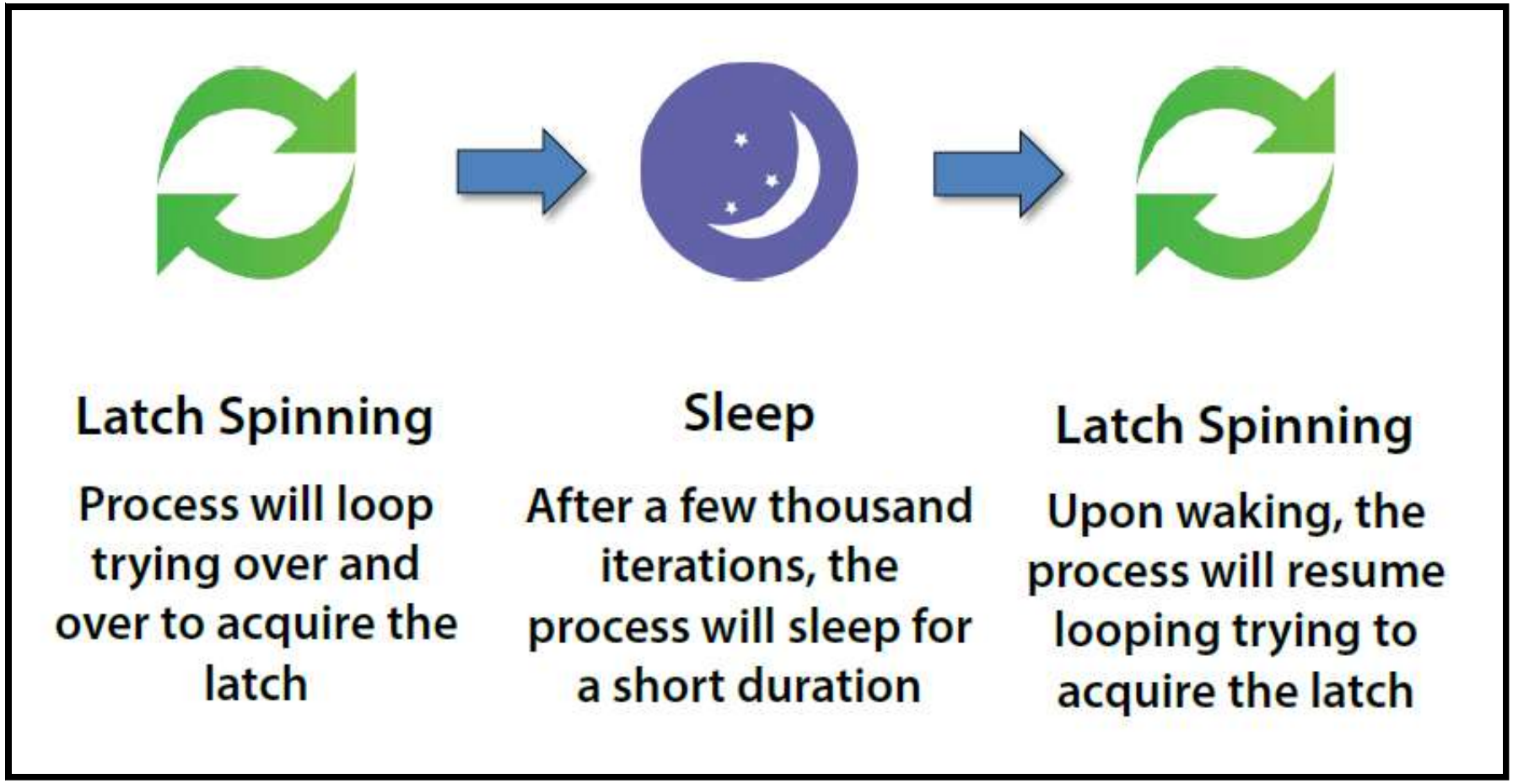
Multiple Processes access the Shared SQL Area



Synchronizing access to the Shared SQL Area



When a Process cannot acquired the `Latch`



Testing

Flush the shared pool to be sure that previous statements don't influence current executions of the query

```
ALTER SYSTEM FLUSH SHARED_POOL;
```

Keeping track of the timing

```
SET TIMING ON
exec WORKLOAD_PROCEDURE;
SET TIMING OFF
```

Bad application

- If there is an application that doesn't use bind variables, the entire database, and other applications that use them, will suffer a drop in performance.
- The "bad" application (the one that doesn't use bind variables) will insert many statements in the library cache.

Avoiding Full Table Scans

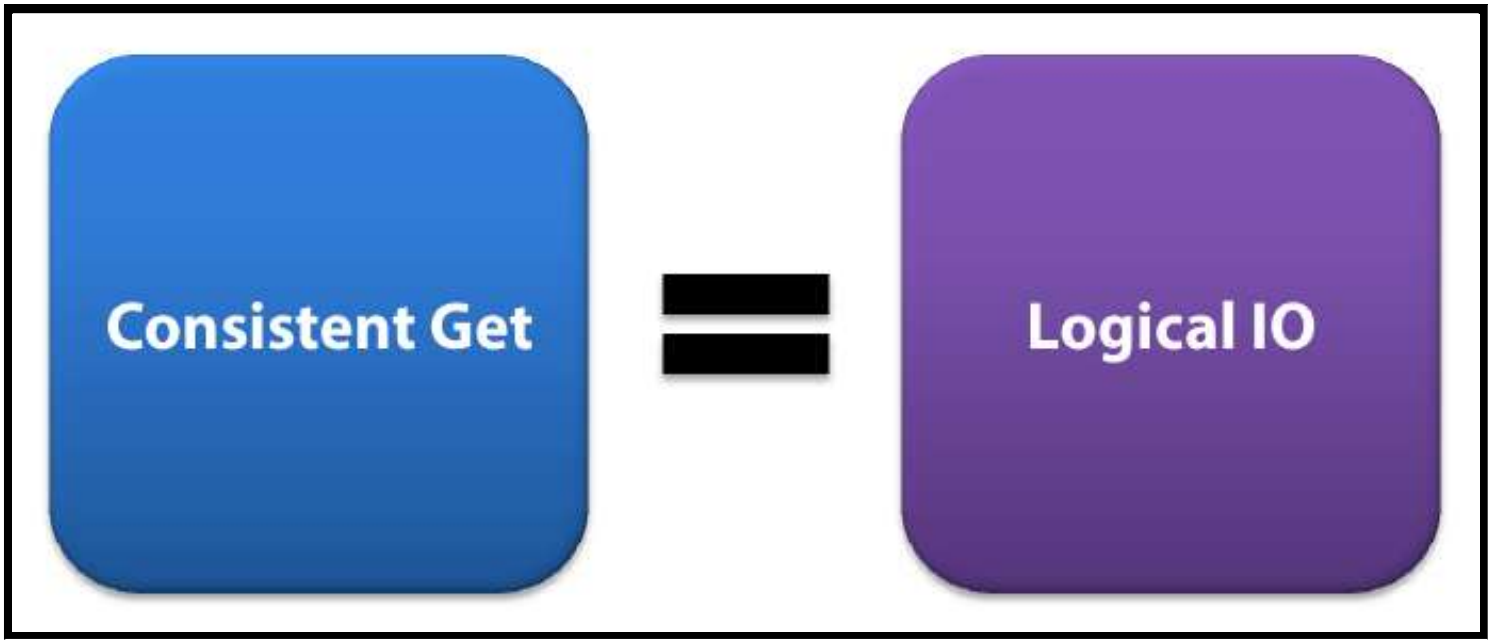
- Problems
- Small Tables
- Data Warehouse
- The High-Water Mark

Logical I/O

When a data block is in the buffer cache

When a data block is not in the buffer cache

Consistent Get



Full Table Scan

- When the database executes a Full Table Scan (FTS) operation, it reads all the database blocks of a table to retrieve the data needed.
- This operation is often undesirable in OLTP databases

Small Tables

- It's better to use a Full Table Scan when we have small tables.
- You should follow this advice because it may be more expensive to read an index entry and the corresponding data instead of reading the full table.

Full Table Scan Problems

The FTS operation has two problems related to performance;

- Many I/O operations are needed to perform the FTS to retrieve the blocks.
- The second flaw is related to the buffer cache.

FTS in Data Warehouse

In data warehouse environments, the use of FTS operations is often preferable. This is because when we have a larger database block, we can read many rows in a block and even subsequent database blocks—in one operation—by setting the parameter `DB_FILE_MULTIBLOCK_READ_COUNT` (at the instance or session level). This parameter controls the number of database blocks read in one I/O operation.

The High-Water Mark

- The High-Water Mark (HWM) is recorded in the segment header block, which indicates the last used block in the segment.
- The HWM is very important in FTS operations—the database will read every single block in the table segments, which are below the HWM, even if they are not used.

Please remember that a DELETE operation won't reset the High-Water Mark, ever!

Testing

Flushing Buffer Cache

```
ALTER SYSTEM FLUSH BUFFER_CACHE;
```

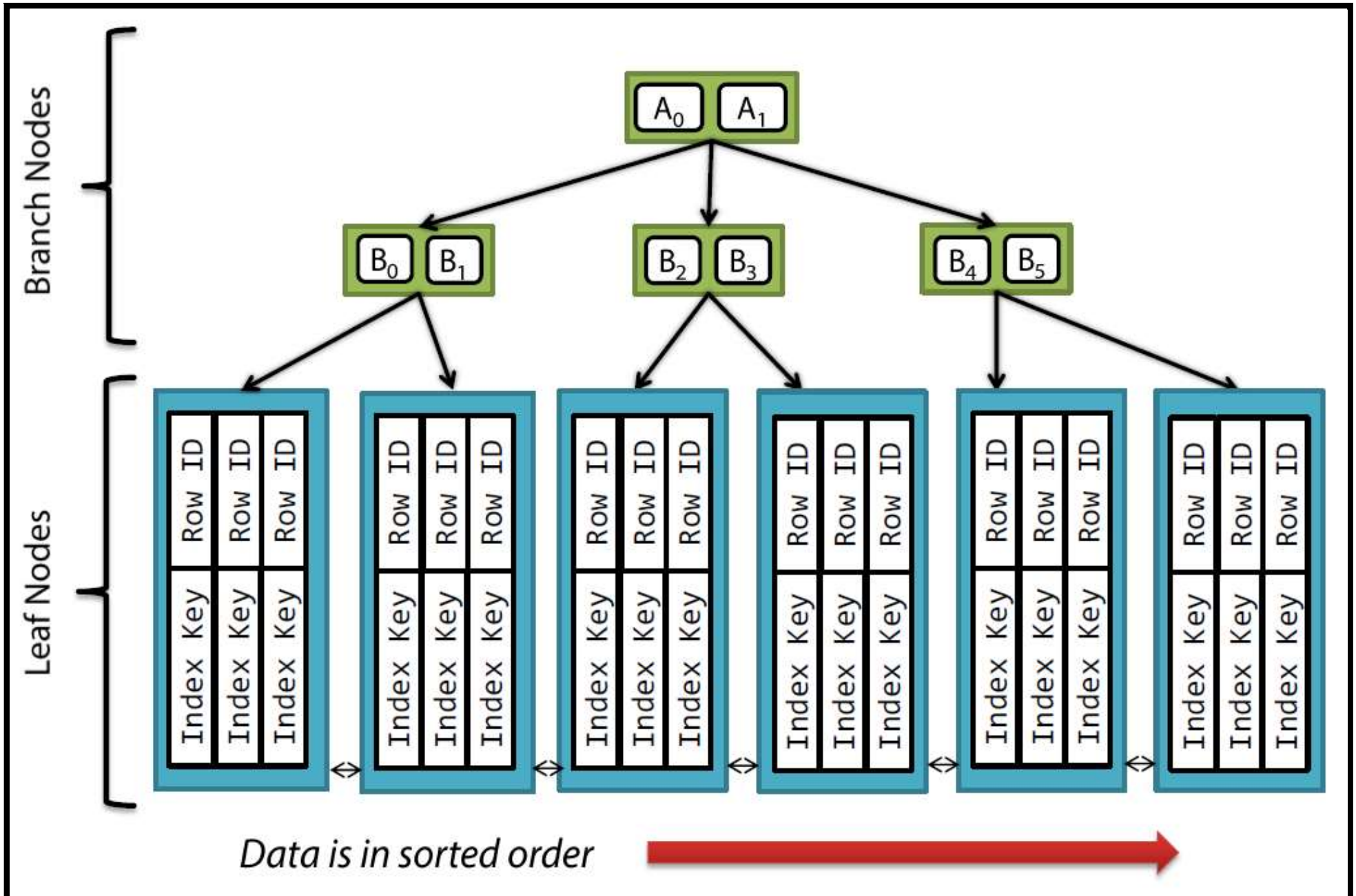
Number of Database Blocks

```
SELECT BLOCKS FROM DBA_TABLES
  WHERE TABLE_NAME = 'MY_TABLE_NAME';
```

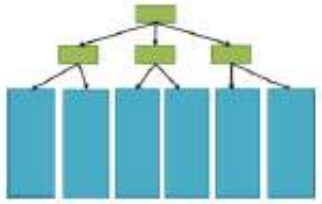
Database Index

- B-Tree
- Bitmap Index

B-Tree


Branch Nodes
Leaf Nodes
A0
A1
B0
B1
B2
B3
B4
B5
Index Key
Row ID
Data is in sorted order

Index Efficiency

	Index (B-Tree)	Table Scan
		
Search Time	**O(log *n*)**	**O(*n*)**
For 1,000,000 items	**19**	**1,000,000**

Bitmap Index

Table Data

Student ID	Gender	Married	Age Range
1001	F	N	18-34
1002	F	Y	35-49
1003	M	Y	35-49
1004	F	Y	35-49
1005	F	Y	18-34
1006	M	Y	62+
1007	F	N	50-61
1008	M	Y	35-49
1009	F	Y	35-49
1010	M	N	35-49
1011	M	Y	35-49
1012	F	N	18-34

Bitmap Index

Female	Married	18-34	35-49	50-62	62+
1	0	1	0	0	0
1	1	1	0	0	0
0	1	0	1	0	0
1	1	0	1	0	0
1	1	1	0	0	0
0	1	0	0	0	1
1	0	0	0	1	0
0	1	0	1	0	0
1	1	0	1	0	0
0	0	0	1	0	0
0	1	0	1	0	0
1	0	1	0	0	0

Bitmap Index Usage

- Useful for columns with a low number of distinct values
- Targeted for data warehouse applications
- Not quite useful in OLTP databases
- Only available in Oracle Enterprise Edition

Using Index

- One of the methods to avoid FTS is indexing.

Create multi-colum indexes

- Try to create multi-column indexes on the attributes of a table you use together in the predicate.
- If the index contains fields of the projection, the database could use the index only to answer the query, without accessing table data.

Index Range Scan

When running an Index Range Scan operation, the database finds the first block of the index containing values that satisfy the predicate and scans the index data block to find all the entries that satisfy the conditions—examining the leaf blocks of the index using the link between the leaves. This operation is very fast.

Index Skip Scan

- The Index Skip Scan operation is a compromise between the Index Range Scan and Fast Full Scan.
- If we perform Index Skip Scan operations, instead, performance will benefit more with a less selective column in the first place of the index.

Oracle might not use index

- We can see that the database will perform an FTS operation. When using the index to access rows in the table will use more resources, because most of the data in the table satisfies the where condition.
- Indexes cannot be used when we compare values with a not equal operator.
- There are other situations which prevent an index from being used:
  - Using a function
  - Searching for NULL values

THE END

- Source code & documentation
- Back to Course Outline