Sperasoft

Your game dev partner

Database Indexes

The Simple Truth

- RDBMS store data only in Trees
- Index is a tree in terms of data structure
- a Table is an Index
- a Clustered Index is a Table itself
- a Non-clustered Index is a copy of data
- all Non-clustered Indexes refer to Clustered one
- all keys in Tree Nodes are always unique

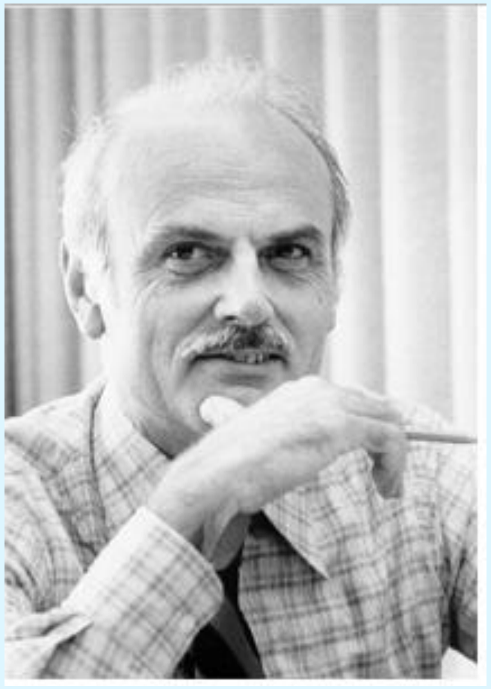
What’s common between

- Oracle Database
- SQL Server
- IBM DB2
- MySQL
- PostgreSQL
- Sybase
- Informix

Relational Database Management Systems

- RDBMS is a type of Database Management System that stores data in the form of related tables
- RDBMS is a Database Management System that is based on the relational model introduced by E.F. Codd
- Data is stored in tables and the relationships among the data are also stored in tables

Edgar Frank “Ted” Codd



- Born on the Isle of Portland in England in 1923
- Died in Florida US in 2003, aged 79
- Mathematic
- Worked for IBM

Edgar Frank “Ted” Codd

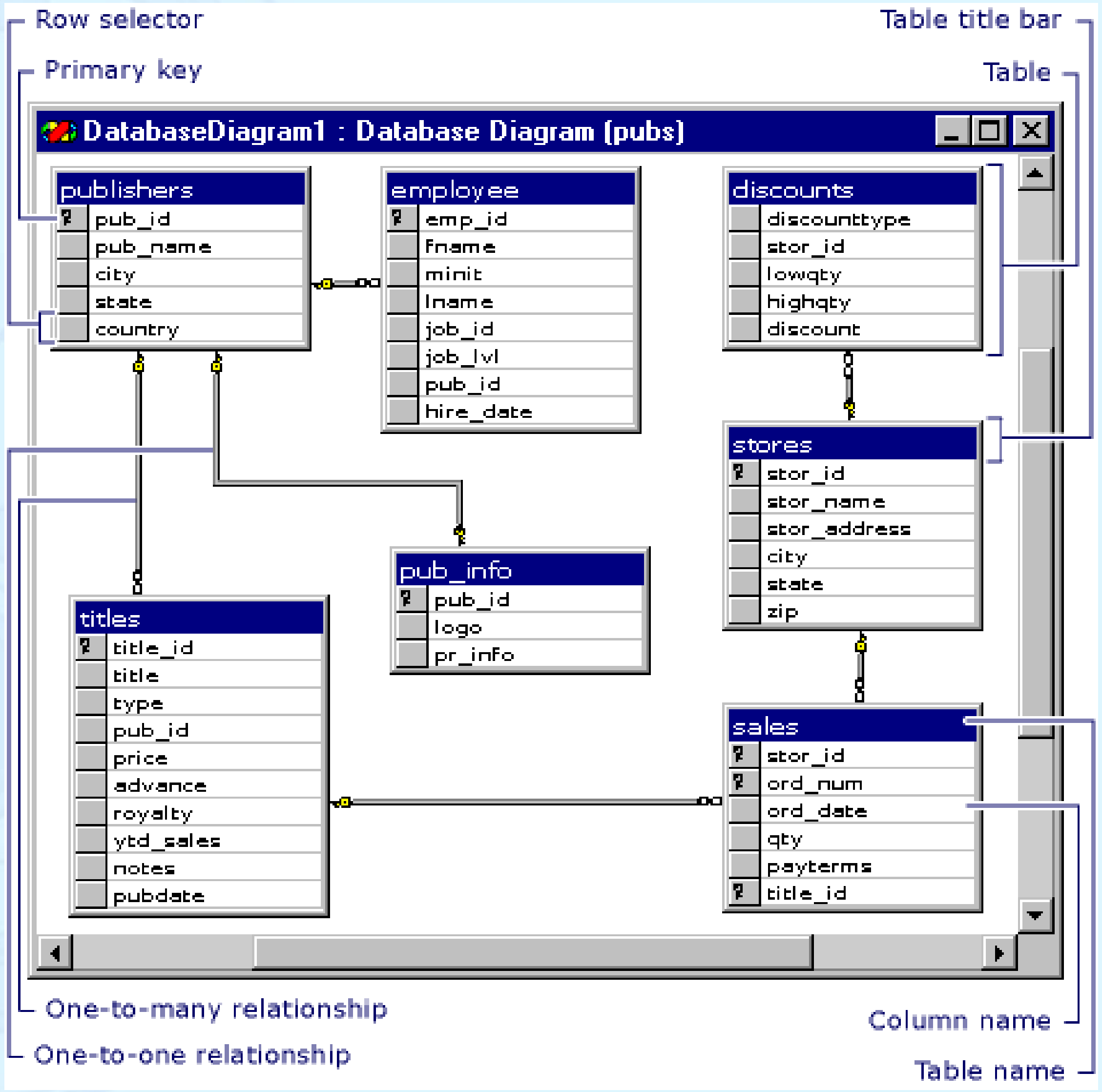
- Introduced “A Relational Model of Data for Large Shared Data Banks” and Alpha database language
- IBM started implementing the Relational model and introduced another language named SEQUEL

Birth of Oracle

- Larry Ellison came up in time with his implementation of Relational model and the language – Oracle Database and SQL
- ANSI started making SQL standard

Relation model briefly

- It’s all about Table Relations

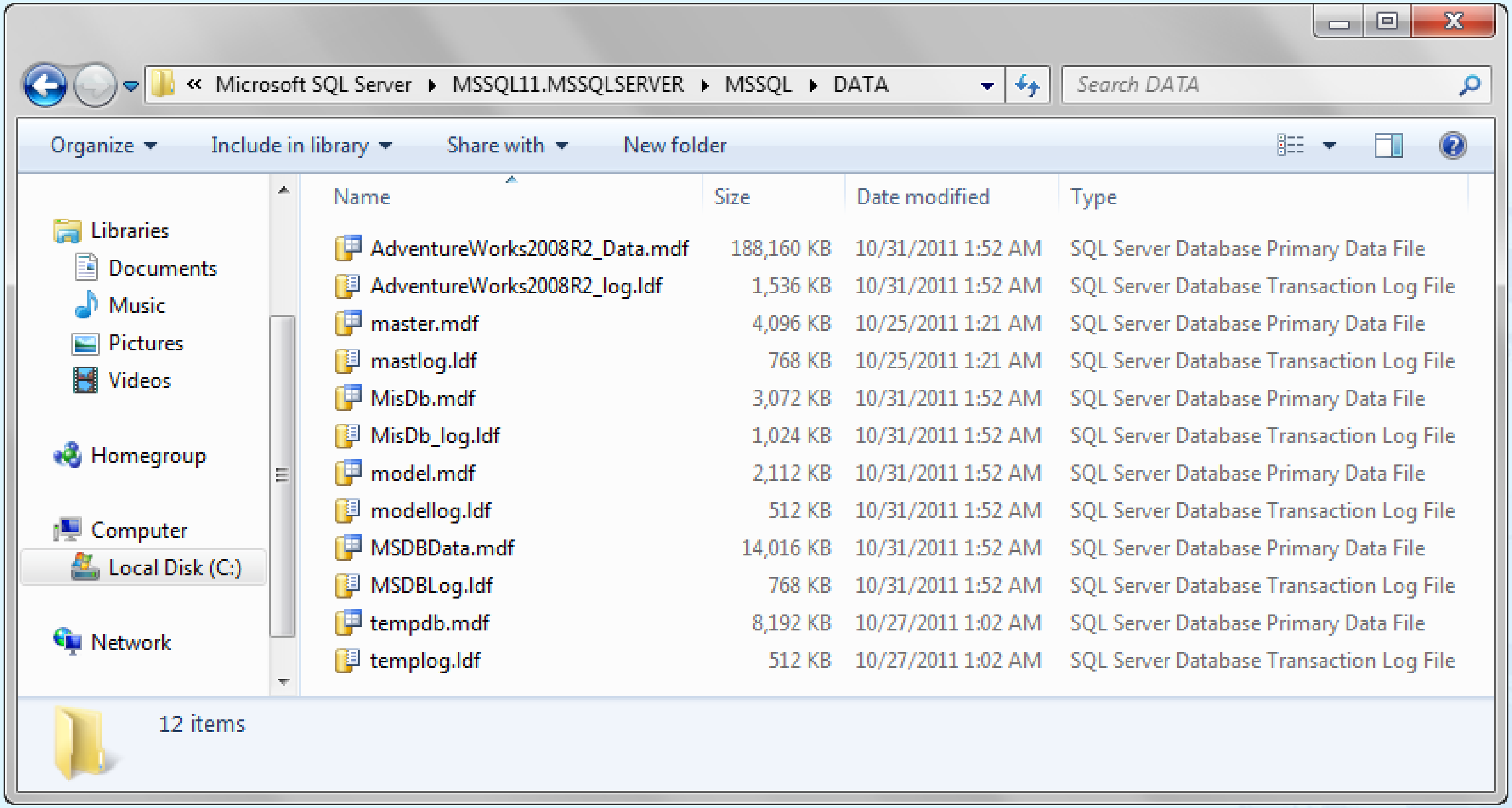


Relation model

- Database contains tables (two dimensional arrays)
- Tables have relationships enforced by Foreign Key constraints (1-to-Many relationship)
- Normalization of tables is a key concept

- That’s why RDBMS are called Relational

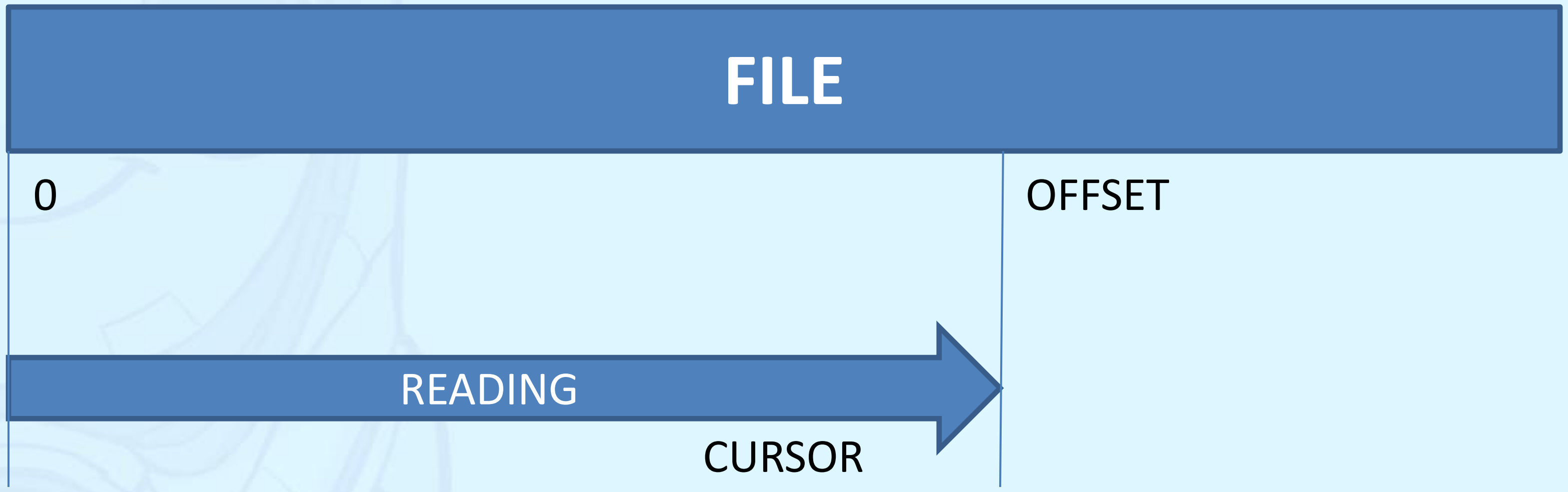
What's database physically



Name	Size	Date modified	Type
AdventureWorks2008R2_Data.mdf	188,160 KB	10/31/2011 1:52 AM	SQL Server Database Primary Data File
AdventureWorks2008R2_log.ldf	1,536 KB	10/31/2011 1:52 AM	SQL Server Database Transaction Log File
master.mdf	4,096 KB	10/25/2011 1:21 AM	SQL Server Database Primary Data File
mastlog.ldf	768 KB	10/25/2011 1:21 AM	SQL Server Database Transaction Log File
MisDb.mdf	3,072 KB	10/31/2011 1:52 AM	SQL Server Database Primary Data File
MisDb_log.ldf	1,024 KB	10/31/2011 1:52 AM	SQL Server Database Transaction Log File
model.mdf	2,112 KB	10/31/2011 1:52 AM	SQL Server Database Primary Data File
modellog.ldf	512 KB	10/31/2011 1:52 AM	SQL Server Database Transaction Log File
MSDBData.mdf	14,016 KB	10/31/2011 1:52 AM	SQL Server Database Primary Data File
MSDBLog.ldf	768 KB	10/31/2011 1:52 AM	SQL Server Database Transaction Log File
tempdb.mdf	8,192 KB	10/27/2011 1:02 AM	SQL Server Database Primary Data File
templog.ldf	512 KB	10/27/2011 1:02 AM	SQL Server Database Transaction Log File

All tables are stored in a file

- Files are flat in nature



What’s actually matter

- What’s the value behind relations?
- What is a database table?
- What is a table index?
- Relations vs How data is stored

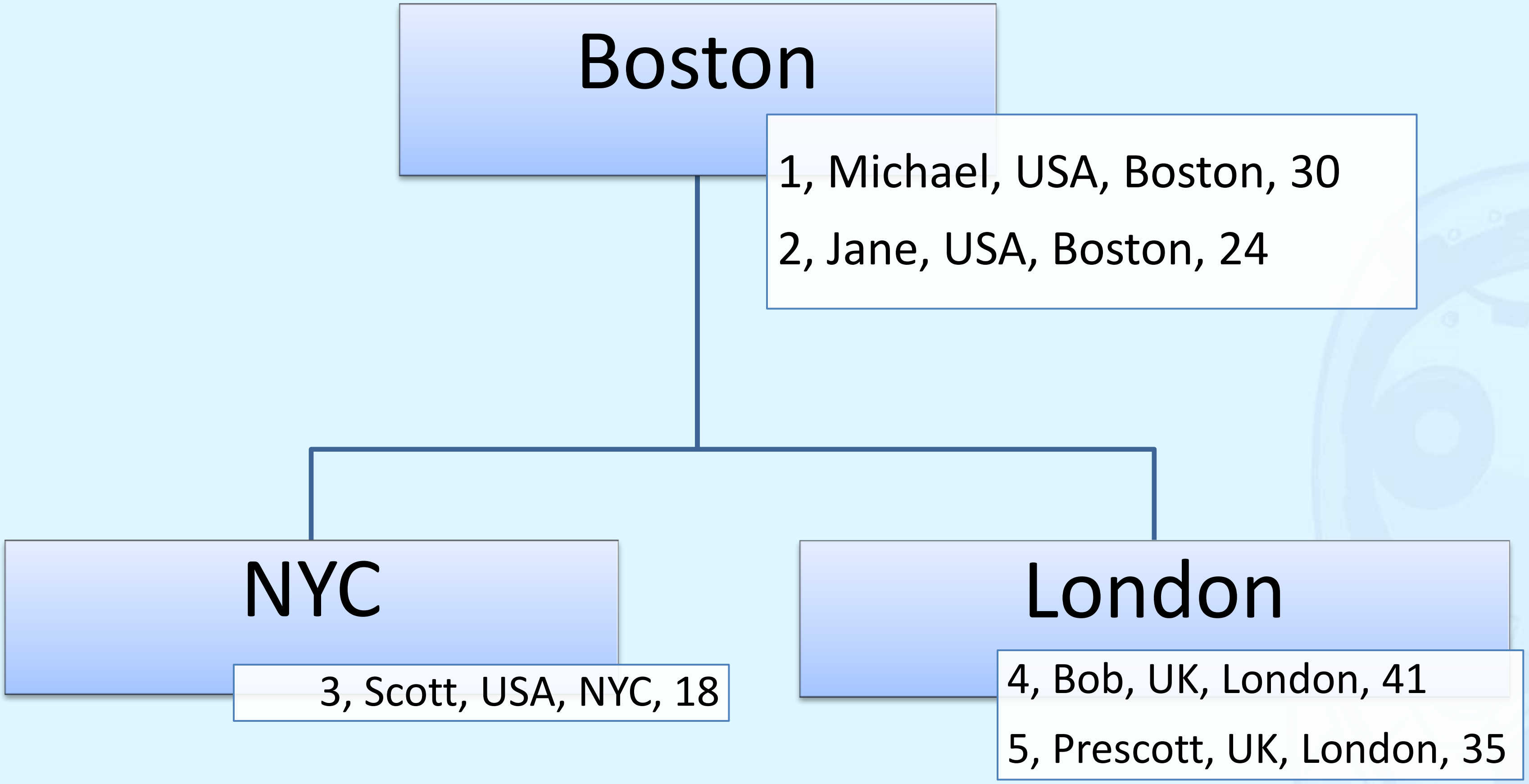
How to handle millions of Users

```
ArrayList<User> users = new ArrayList<User>();
```

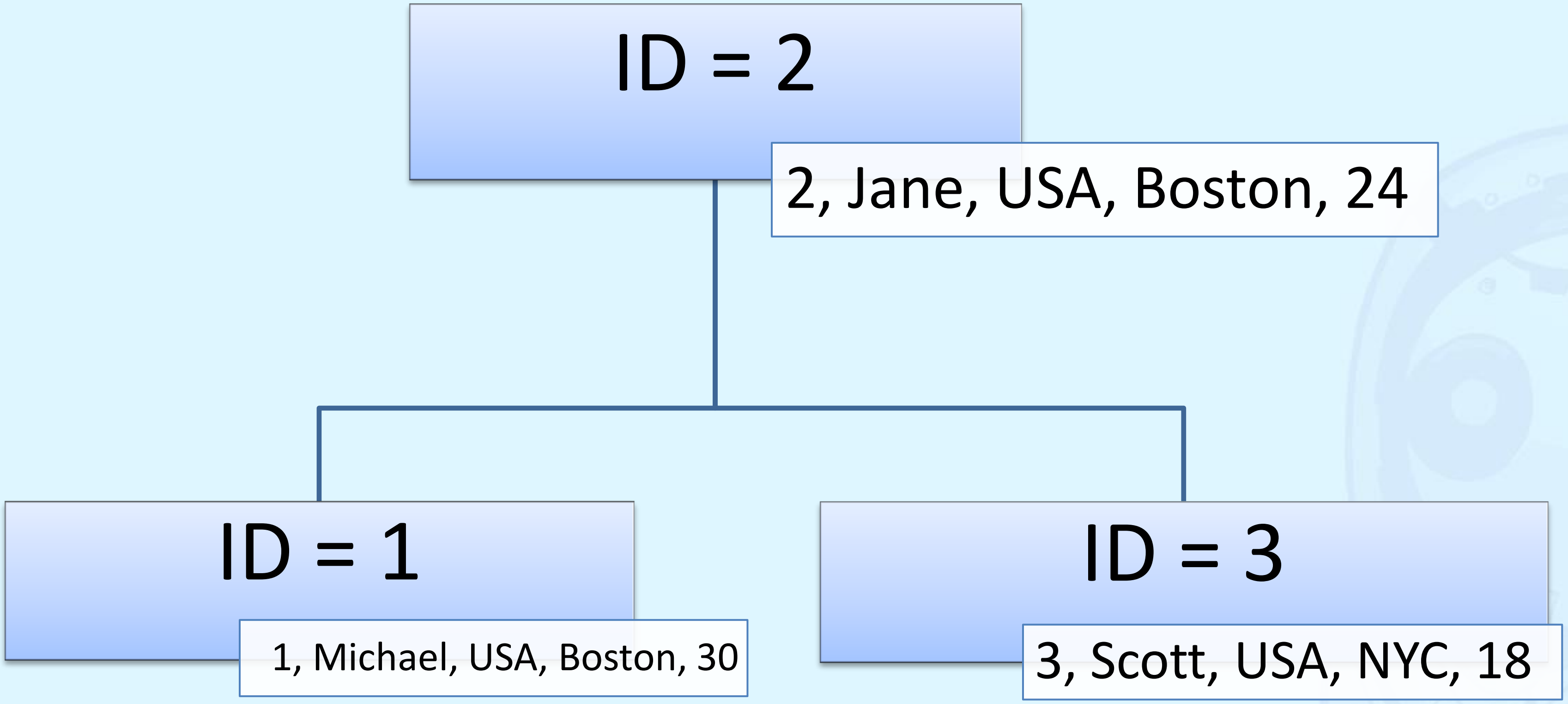
- Such array seems to be a table
- How to find Users from Boston faster?

Id	User Name	Country	City	Age
1	Michael	USA	Boston	30
2	Jane	USA	Boston	24
3	Scott	USA	NYC	18
4	Bob	UK	London	41
5	Prescott	UK	London	35

Index is a tree



Can replace an initial array with Index



What’s important to note

- Key values in a Key node should be unique
- Otherwise Trees do not work

Returning to our sheep

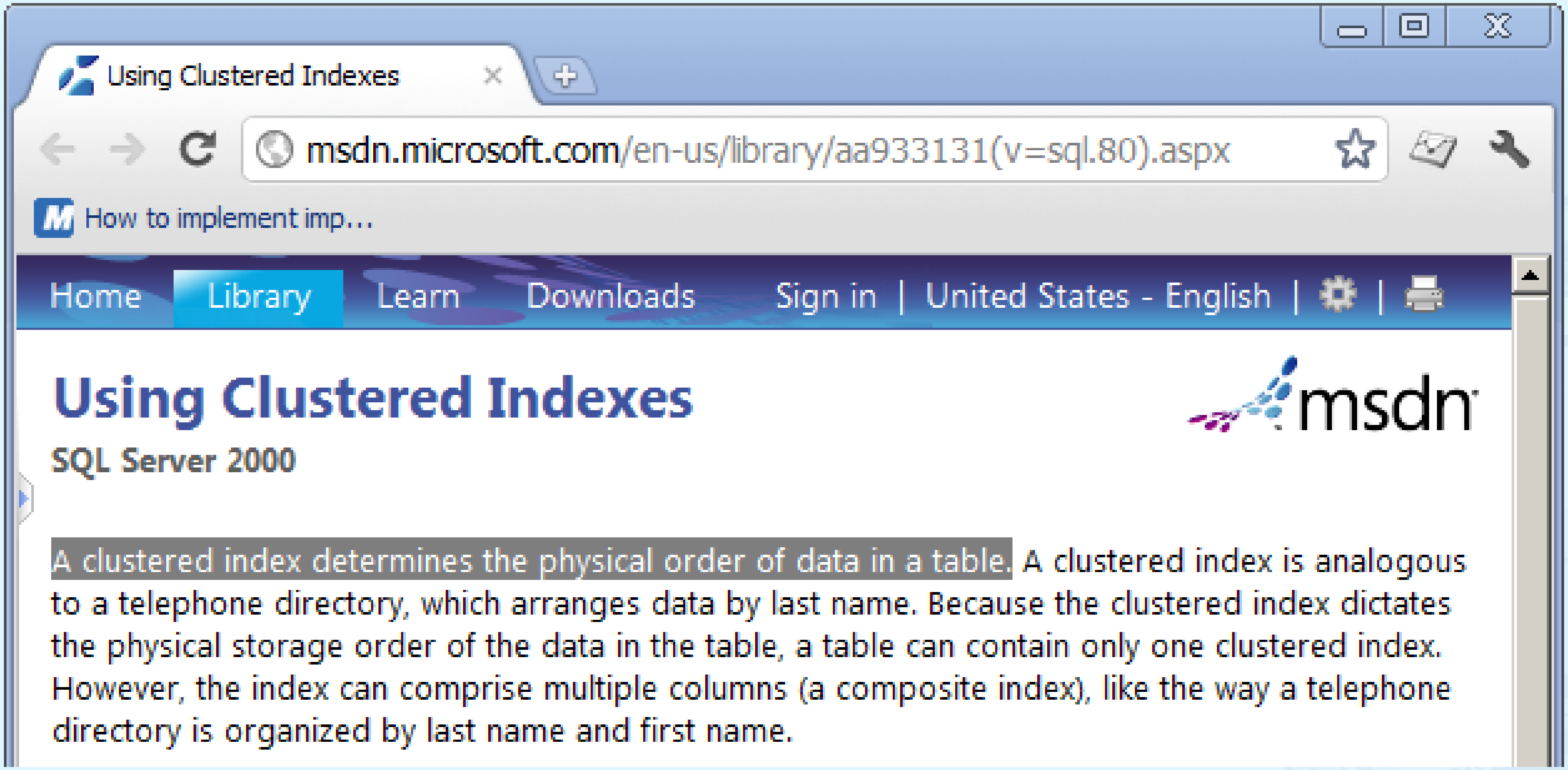
- Indexes are Trees in terms of data structure
- Trees are suitable to store any array of data to make search faster

Balanced Trees

- All RDBMS store data as Balanced Trees
- The concrete implementation of B-Tree could differ from vendor to vendor

- It means the only way to store data is Tree
- No exceptions here - table is a tree, index is a tree

What's a Clustered Index


Using Clustered Indexes
msdn.microsoft.com/en-us/library/aa933131(v=sql.80).aspx
How to implement imp...
Home
Library
Learn
Downloads
Sign in | United States - English | |

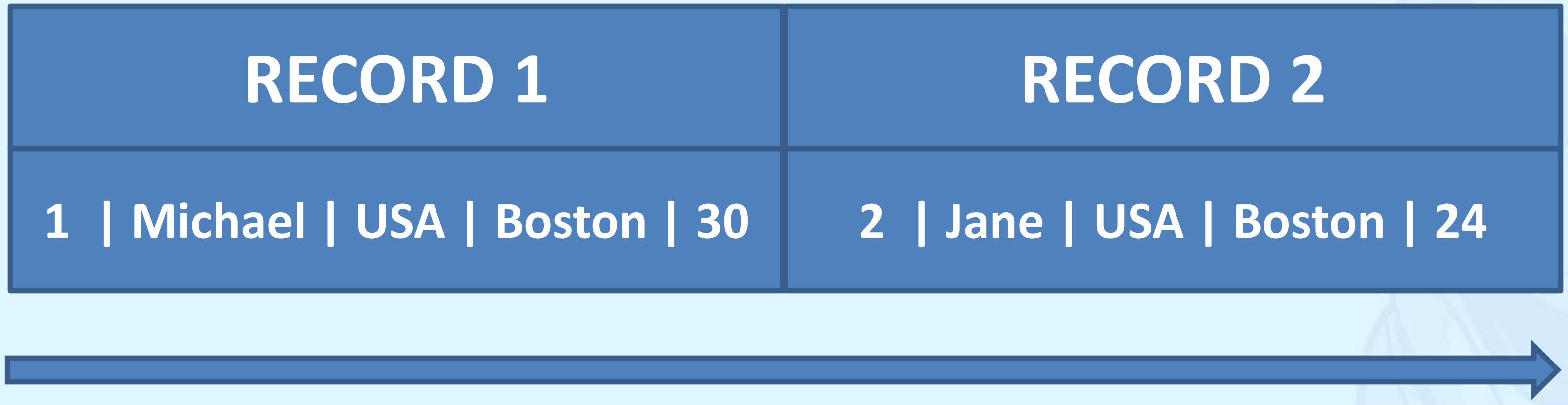
Using Clustered Indexes

SQL Server 2000

A clustered index determines the physical order of data in a table. A clustered index is analogous to a telephone directory, which arranges data by last name. Because the clustered index dictates the physical storage order of the data in the table, a table can contain only one clustered index. However, the index can comprise multiple columns (a composite index), like the way a telephone directory is organized by last name and first name.

The clustered index storage

- The next record in Clustered Index is always stored after the previous one



What SQL allows us to do

- Clustered Indexes
- Non-clustered indexes
- Both could be unique and non-unique
- Table can be without any indexes

- How is that comply with how data is actually stored?

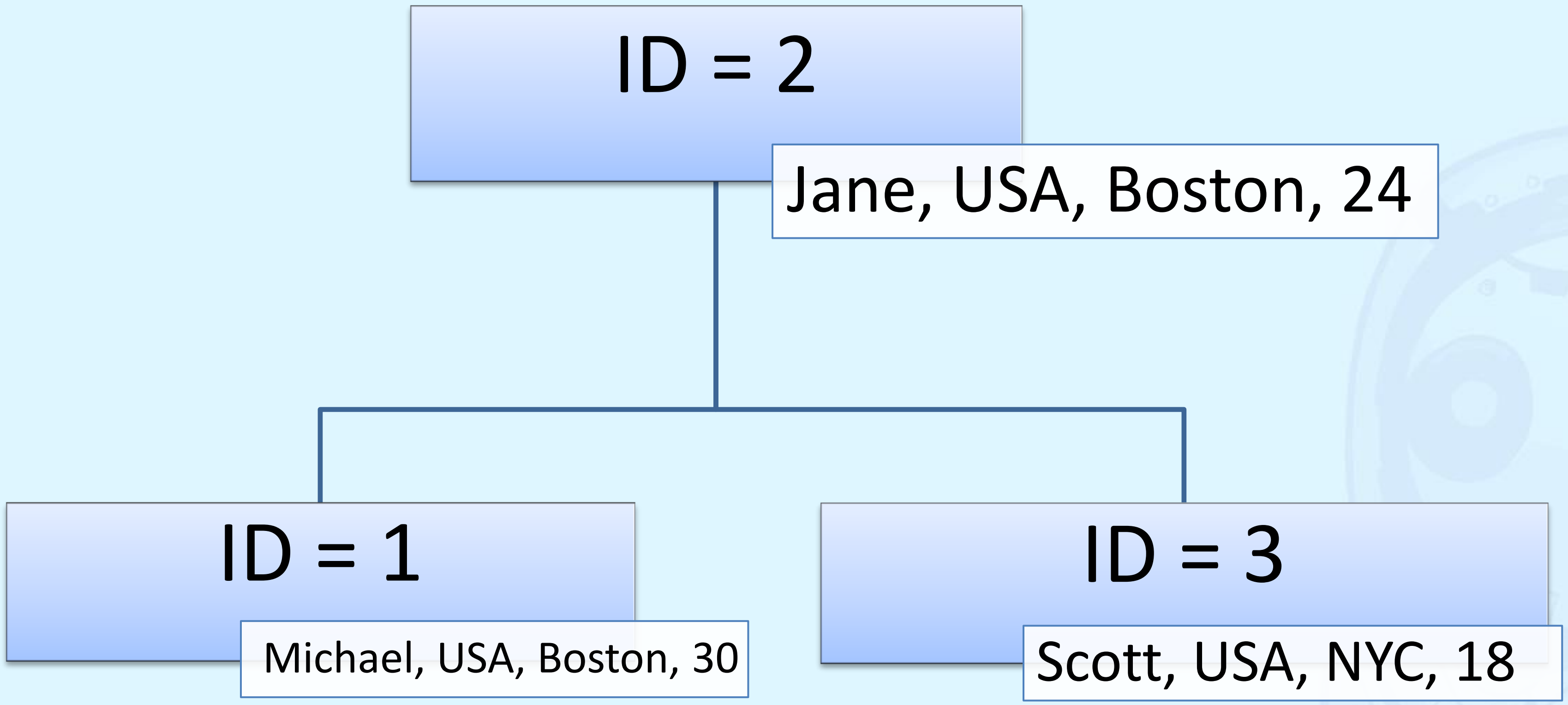
Clustered Indexes

- Unique and non-unique
- CREATE CLUSTERED INDEX [name] ON [table_name] ([column1], [column2])
- CREATE UNIQUE CLUSTERED INDEX [name] ON [table_name] ([column1], [column2])

Nonclustered Indexes

- Unique and non-unique
- CREATE NONCLUSTERED INDEX [name] ON [table_name] ([column1], [column2])
- CREATE UNIQUE NONCLUSTERED INDEX [name] ON [table_name] ([column1], [column2])

Unique Clustered Index



Non-unique Clustered Index

- We know Key values should be unique
- How RDBMS resolves this problem?

Non-unique Clustered Index

- SQL Server adds 4-byte uniquifier to each duplicated key value

- Algorithms could differ from vendor to vendor
- But the principle is the same – add something to make them unique

Clustered Indexes

- Just omitting Unique keyword makes Key values bigger (why it’s bad realize later)

- The simple truth is that Each table should have Clustered Index
- The Clustered Index should be always Unique

- The situations when its not so should be exceptional

Tables without Clustered Index

- Such tables are called Heap Tables
- How are they stored in database if they do not have a Key value specified?

No magic over here

- Heap Tables are also stored in Trees
- What's in a Key value for Tables without Clustered Index?
- The value called RID
- the unique identifier which refers to the physical location of the record in a file

Why Heap Tables are so bad

- There is no meaningful data in Keys
- Table records are not stored physically in Keys' order

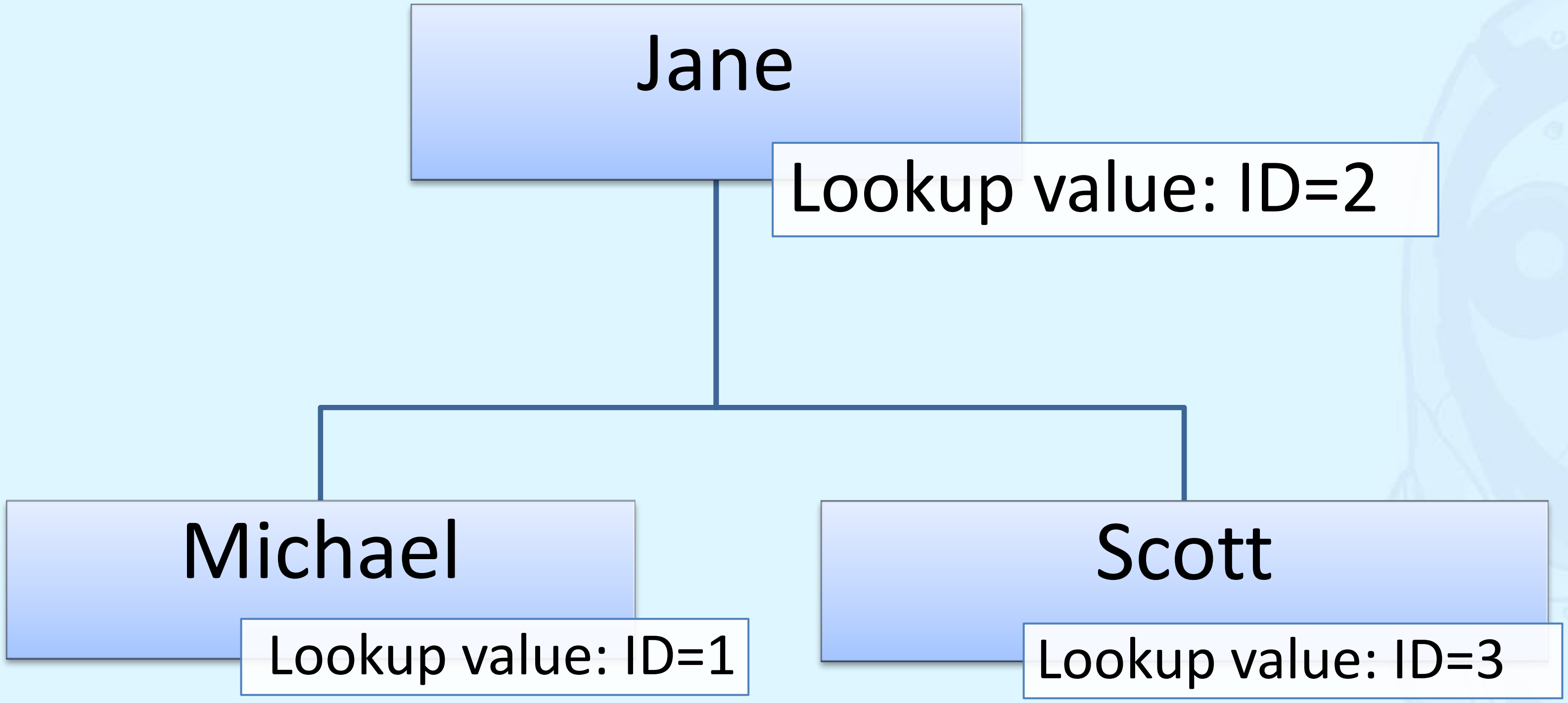
Non-clustered Indexes

- Clustered Index has the actual data columns in Leaf-nodes
- What’s in Leaf-node of Non-clustered index?

- Remember that Non-clustered Indexes are duplicated data

Non-clustered Index

- Leaf-nodes contain the lookup values
- Lookup value is Clustered Index’s Key

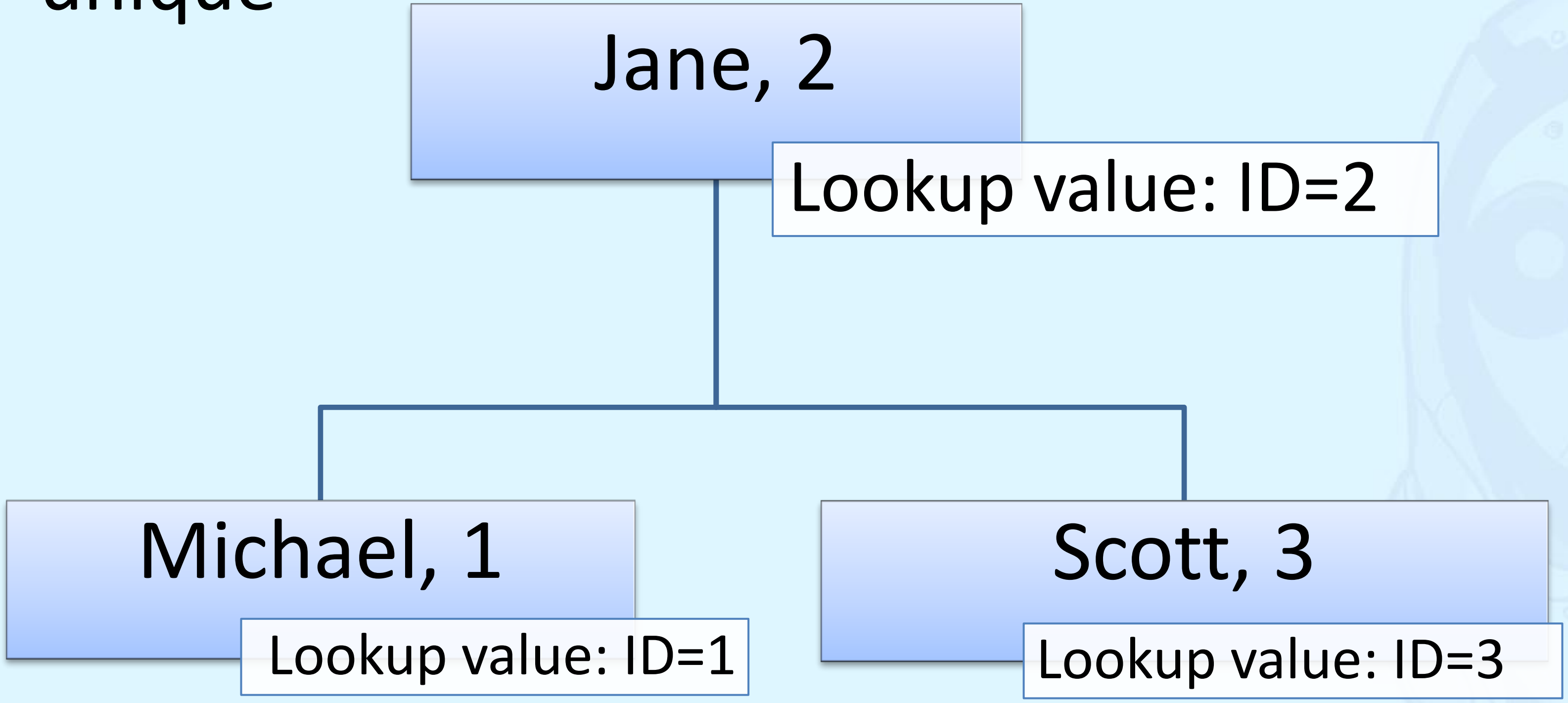


Non-unique Non-clustered Index

- We know Key values should be unique
- How non-clustered index's key becomes unique?

Non-unique Non-clustered Index

- SQL Server adds Clustered Index Key value to Non-clustered Index Key value to make it unique



How indexes are used (1)

- from SELECT statement the WHERE condition is taken
- based on the Columns in WHERE we know what columns we search by
- look through available indexes trying to find the appropriate one, starting from Clustered
- found out non-clustered index which fits best

How indexes are used (2)

- get the needed Node in Non-clustered index
- get the Lookup value from that Node
- use that lookup value to find a record in Clustered index
- get selected columns from Clustered index (table itself)

Sample 1

- Unique Clustered Index on Id column
- Non-unique Non-clustered Index on City column

- Select UserName from tbl where City = ‘Boston’

Id	User Name	Country	City	Age
1	Michael	USA	Boston	30
2	Jane	USA	Boston	24
3	Scott	USA	NYC	18
4	Bob	UK	London	41
5	Prescott	UK	London	35

Sample 2

- Unique Clustered Index on Id column
- Non-unique Non-clustered Index on City column

- Select Id from tbl where City = ‘Boston’

Id	User Name	Country	City	Age
1	Michael	USA	Boston	30
2	Jane	USA	Boston	24
3	Scott	USA	NYC	18
4	Bob	UK	London	41
5	Prescott	UK	London	35

Sample 3

- Unique Clustered Index on Id column
- Non-unique Non-clustered Index on City column

- Select UserName from tbl where City = ‘Boston’ select should not go to Clustered Index

Id	User Name	Country	City	Age
1	Michael	USA	Boston	30
2	Jane	USA	Boston	24
3	Scott	USA	NYC	18
4	Bob	UK	London	41
5	Prescott	UK	London	35

Sample 1

- Unique Clustered Index on Id, UserName column
- Select Id from tbl where City = ‘Boston’ and UserName = ‘Michael’

- What columns Non-unique Non-clustered Index would include?

Id	User Name	Country	City	Age
1	Michael	USA	Boston	30
2	Jane	USA	Boston	24
3	Scott	USA	NYC	18
4	Bob	UK	London	41
5	Prescott	UK	London	35