KG REDDY College of Engineering & Technology

Database management concepts

- Database Management Systems (DBMS)
  - An example of a database (relational)
  - Database schema (e.g. relational)
  - Data independence
  - Architecture of a DBMS
  - Types of DBMS
  - Basic DBMS types
  - Retrieving and manipulating data: query processing
  - Database views
- Data integrity
- Client-Server architectures
- Knowledge Bases and KBS (and area of AI)

- DBMS tasks:
  - Managing large quantity of structured data
  - Efficient retrieval and modification: query processing and optimization
  - Sharing data: multiple users use and manipulate data
  - Controlling the access to data: maintaining the data integrity
- An example of a database (relational):
  - Relations (tables)
  - Attributes (columns)
  - Tuples (rows)
  - Example query: Salesperson='Mary' AND Price>100.

Relation ITEM

Item#	ItemName	Quantity	Location
123	pump	25	A23.2
235	saw	42	B3.9
589	hose	110	A23.5
601	ladder	12	B14.6
...	...	...	...

Relation SUPPLIES

Item#	Supplier#
123	23
235	23
589	99
601	6
...	...

Relation SALES

Item#	Salesperson	Price
601	Sam	169.95
123	Sam	99.95
589	Mary	24.98
601	John	169.95
123	Mary	99.95
601	Mary	169.95
235	John	25.49
...	...	...

Relation SUPPLIER

Supplier#	City	Phone
6	Albany	518-555-1234
23	Troy	518-555-4321
48	Schenectady	518-555-6789
99	New York	201-555-9876
...	...	...

Figure 9.1 The inventory relational database

- Database schema (e.g. relational):
  - Names and types of attributes
  - Addresses
  - Indexing
  - Statistics
  - Authorization rules to access data etc.
- Data independence: separation of the physical and logical data
  - Particularly important for distributed systems
  - The mapping between them is provided by the schema
- Architecture of a DBMS - three levels: external, conceptual and internal schema
- Types of DBMS
  - The data structures supported: tables (relational), trees, networks, objects
  - Type of service provided: high level query language, programming primitives

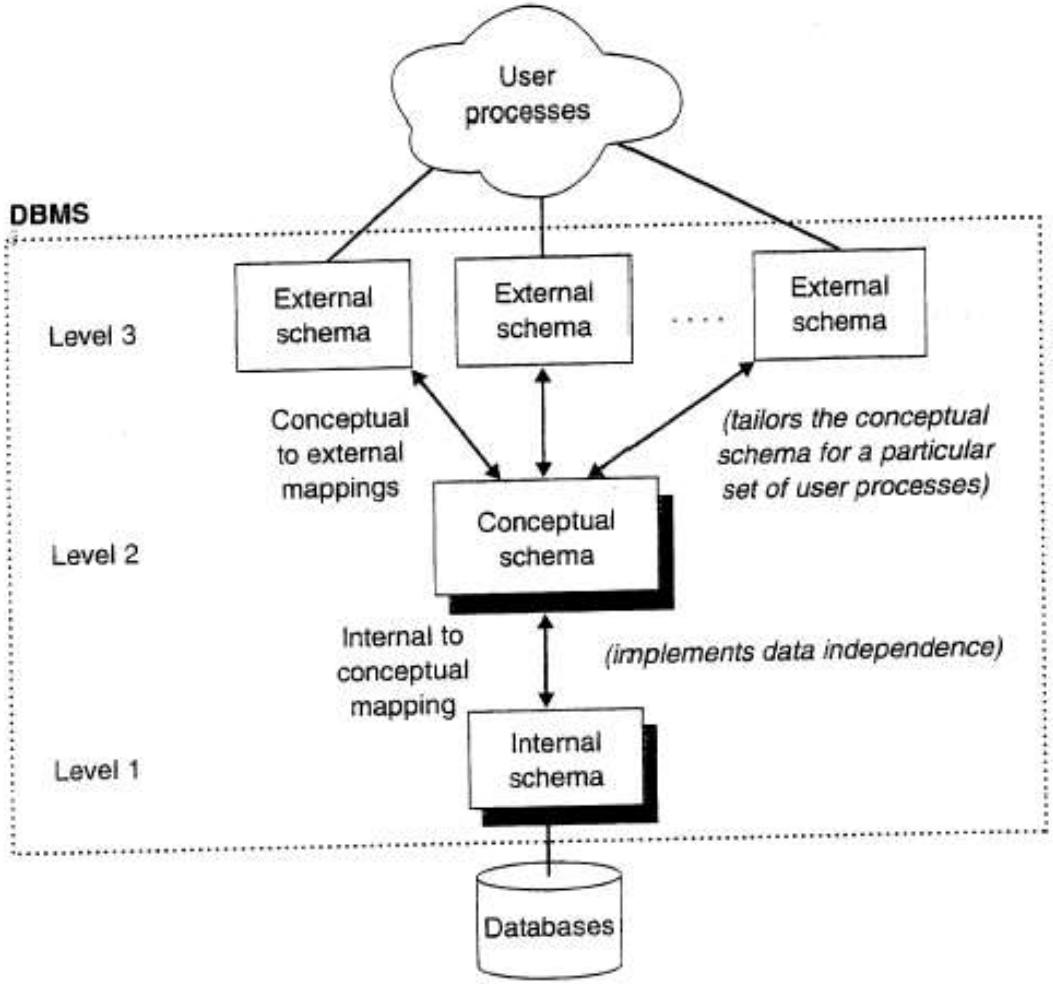


Figure 9.2 The three levels in a DBMS. *Source:* D. Tsichritzis and A. Klug, eds., *The ANSI/X3/SPARC DBMS Framework* (Montvale, NJ: AFIPS Press, 1978).

Basic DBMS types

- Linear files
  - Sequence of records with a fixed format usually stored on a single file
  - Limitation: single file
  - Example query: Salesperson='Mary' AND Price>100
- Hierarchical structure
  - Trees of records: one-to-many relationships
  - Limitations:
    - Requires duplicating records (e.g. many-to-many relationship)
    - Problems when updated
    - Retrieval requires knowing the structure (limited data independence): traversing the tree from top to bottom using a procedural language
- Network structure: similar to the hierarchical database with the implementation of many-to-many relationships
- Relational structure
- Object-Oriented structure
  - Objects (collection of data items and procedures) and interactions between them.
  - Is this really a new paradigm, or a special case of network structure?
  - Separate implementation vs. implementation on top of a RDBMS

Relational structure

- Relations, attributes, tuples
- Primary key (unique combination of attributes for each tuple)
- Foreign keys: relationships between tuples (many-to-many).

  Example: SUPPLIES defines relations between ITEM and SUPPLIER tuples.
- Advantages: many-to-many relationships, high level declarative query language (e.g. SQL)
- SQL example (retrieve all items supplied by a supplier located in Troy):

```
SELECT ItemName
FROM ITEM, SUPPLIES, SUPPLIER
WHERE SUPPLIER.City = "Troy" AND
    SUPPLIER.Supplier# = SUPPLIES.Supplier# AND
    SUPPLIES.Item# = ITEM.Item#
```

- Programming language interfaces: including SQL queries in the code

Retrieving and manipulating data: query processing

- Parsing and validating a query: data dictionary - a relation listing all relations and relations listing the attributes
- Plans for computing the query: list of possible way to execute the query, estimated cost for each. Example:

```
SELECT ItemNames, Price
FROM ITEM, SALES
WHERE SALES.Item# = ITEM.Item# AND Salesperson="Mary"
```

- Index: B-tree index, drawbacks - additional space, updating; indexing not all relations (e.g. the keys only)
- Estimating the cost for computing a query: size of the relation, existence/size of the indices. Example: estimating Attribute=value with a given number of tuples and the size of the index.
- Query optimization: finding the best plan (minimizing the computational cost and the size of the intermediate results), subsets of tuples, projection and join.
- Static and dynamic optimization

Database views

- Creating user defined subsets of the database
- Improving the user interface
- Example:

```
CREATE VIEW MarySales(ItemName,Price)
AS SELECT ItemName, Price
FROM ITEM, SALES
WHERE ITEM.Item#=SALES.Item# AND Salesperson="Mary"
```

Then the query:

```
SELECT ItemName
FROM MarySales
WHERE Proce>100
```

translates to:

```
SELECT ItemName
FROM ITEM, SALES
WHERE ITEM.Item#=SALES.Item# AND Salesperson="Mary" AND Price>100
```

Data integrity

Integrity constraints: semantic conditions on the data

- Individual constraints on data items
- Uniqueness of the primary keys
- Dependencies between relations

Concurrency control

- Steps in executing a query
- Concurrent users of the database, interfering the execution of one query by another
- Transaction: a set of operations that takes the database from one consistent state to another
- Solving the concurrency control problem: making transactions atomic operations (one at a time)
- Concurrent transactions: serializability theory (two-phase locking), read lock (many), write lock (one).
- Serializible transactions: first phase - accumulating locks, second phase - releasing locks.
- Deadlocks: deadlock detection algorithms.
- Distributed execution problems:
  - release a lock at one node (all locks accumulated at the other node?)
  - strict two-phase locking

KG REDDY College of Engineering & Technology

The Transaction Model

Primitive	Description
BEGIN_TRANSACTION	Make the start of a transaction
END_TRANSACTION	Terminate the transaction and try to commit
ABORT_TRANSACTION	Kill the transaction and restore the old values
READ	Read data from a file, a table, or otherwise
WRITE	Write data to a file, a table, or otherwise

- Examples of primitives for transactions.

The Transaction Model

```
BEGIN_TRANSACTION
  reserve WP -> JFK;
  reserve JFK -> Nairobi;
  reserve Nairobi -> Malindi;
END_TRANSACTION
```

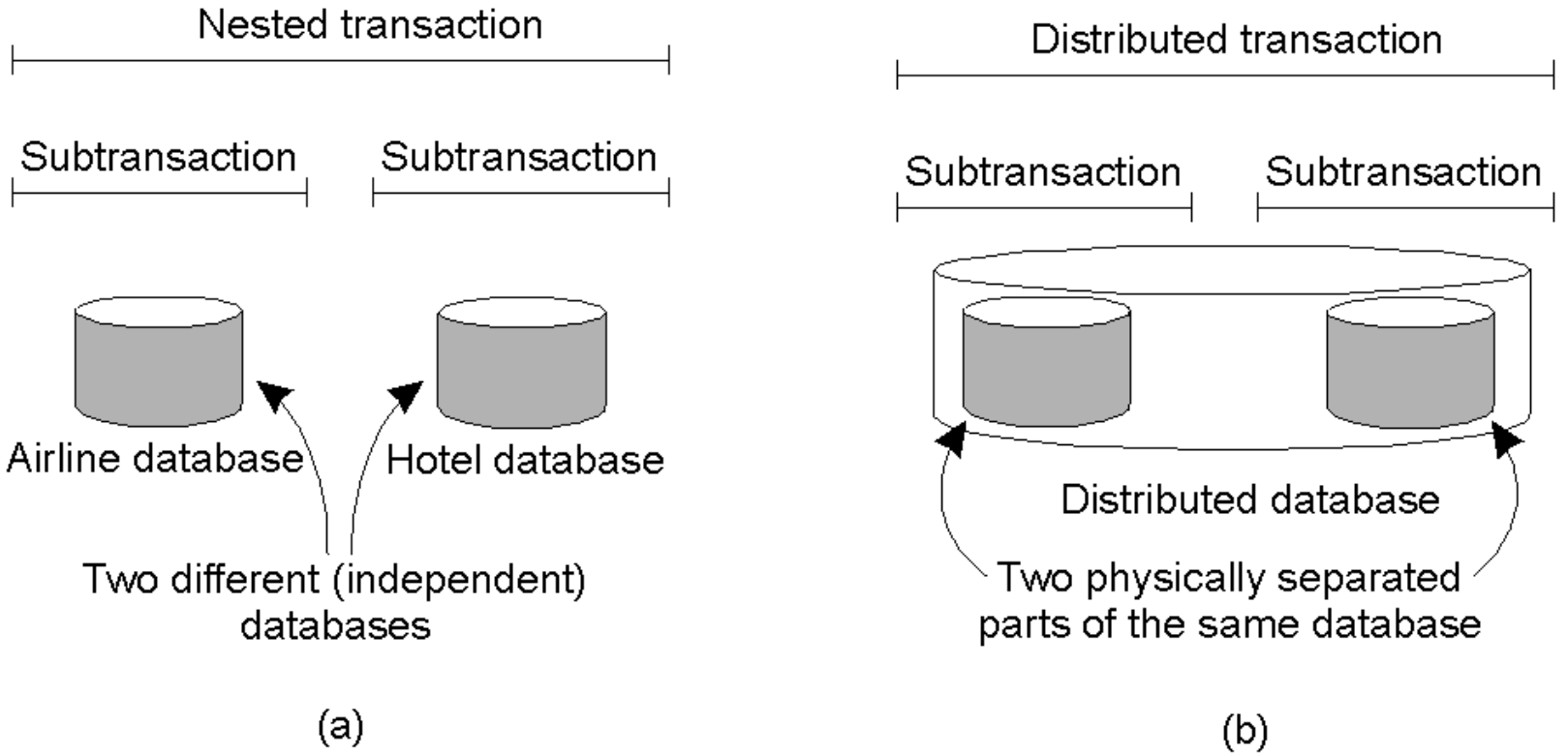
(a)

```
BEGIN_TRANSACTION
  reserve WP -> JFK;
  reserve JFK -> Nairobi;
  reserve Nairobi -> Malindi full =>
ABORT_TRANSACTION
```

(b)

a) Transaction to reserve three flights commits
b) Transaction aborts when third flight is unavailable

Distributed Transactions



a) A nested transaction

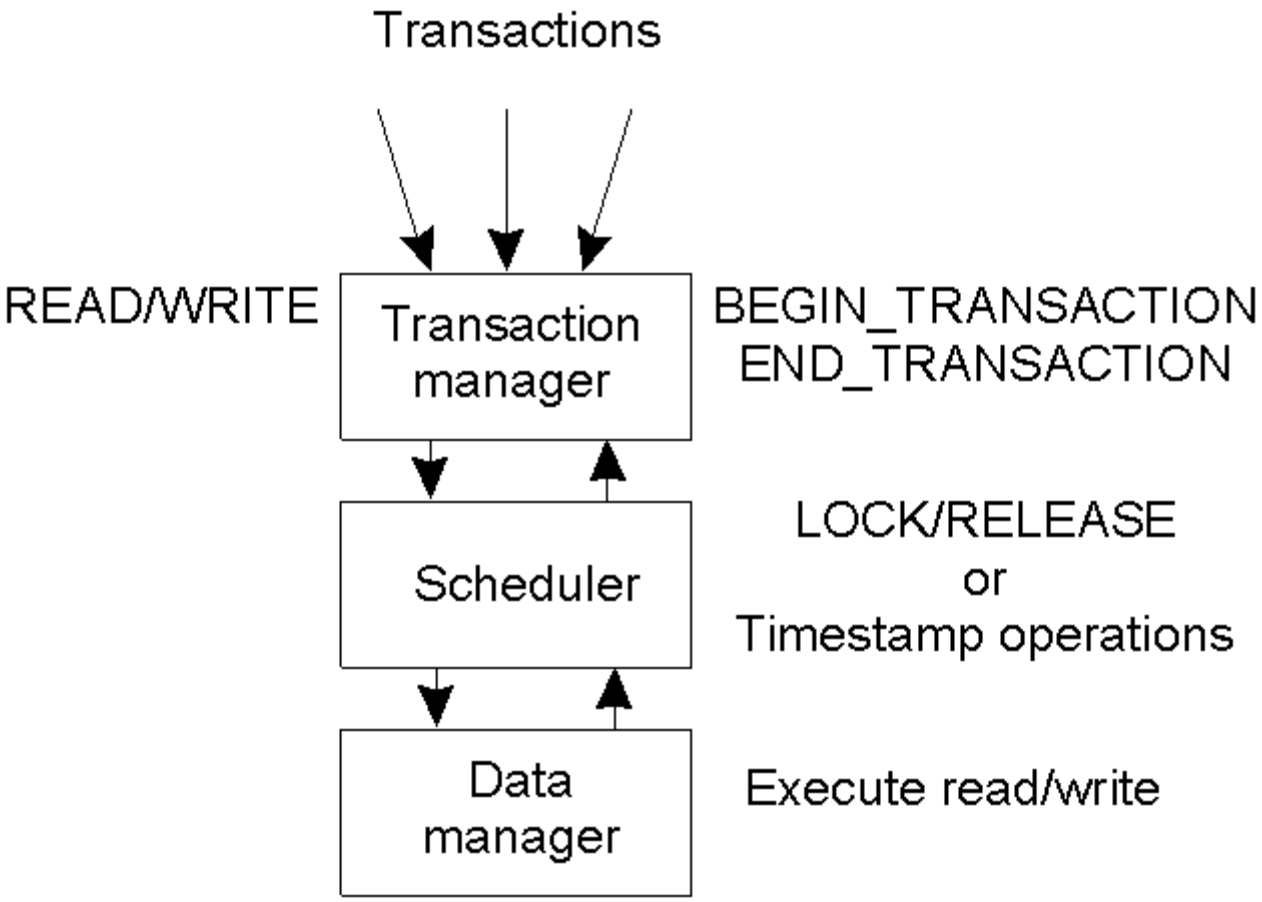
b) A distributed transaction

Writeahead Log

x = 0;	Log	Log	Log
y = 0;			
BEGIN_TRANSACTION;			
x = x + 1;	[x = 0 / 1]	[x = 0 / 1]	[x = 0 / 1]
y = y + 2		[y = 0/2]	[y = 0/2]
x = y * y;			[x = 1/4]
END_TRANSACTION;			
(a)	(b)	(c)	(d)

- a) A transaction
- b) – d) The log before each statement is executed

Concurrency Control (1)



- General organization of managers for handling transactions.

Serializability

```
BEGIN_TRANSACTION
  x = 0;
  x = x + 1;
END_TRANSACTION
```

(a)

```
BEGIN_TRANSACTION
  x = 0;
  x = x + 2;
END_TRANSACTION
```

(b)

```
BEGIN_TRANSACTION
  x = 0;
  x = x + 3;
END_TRANSACTION
```

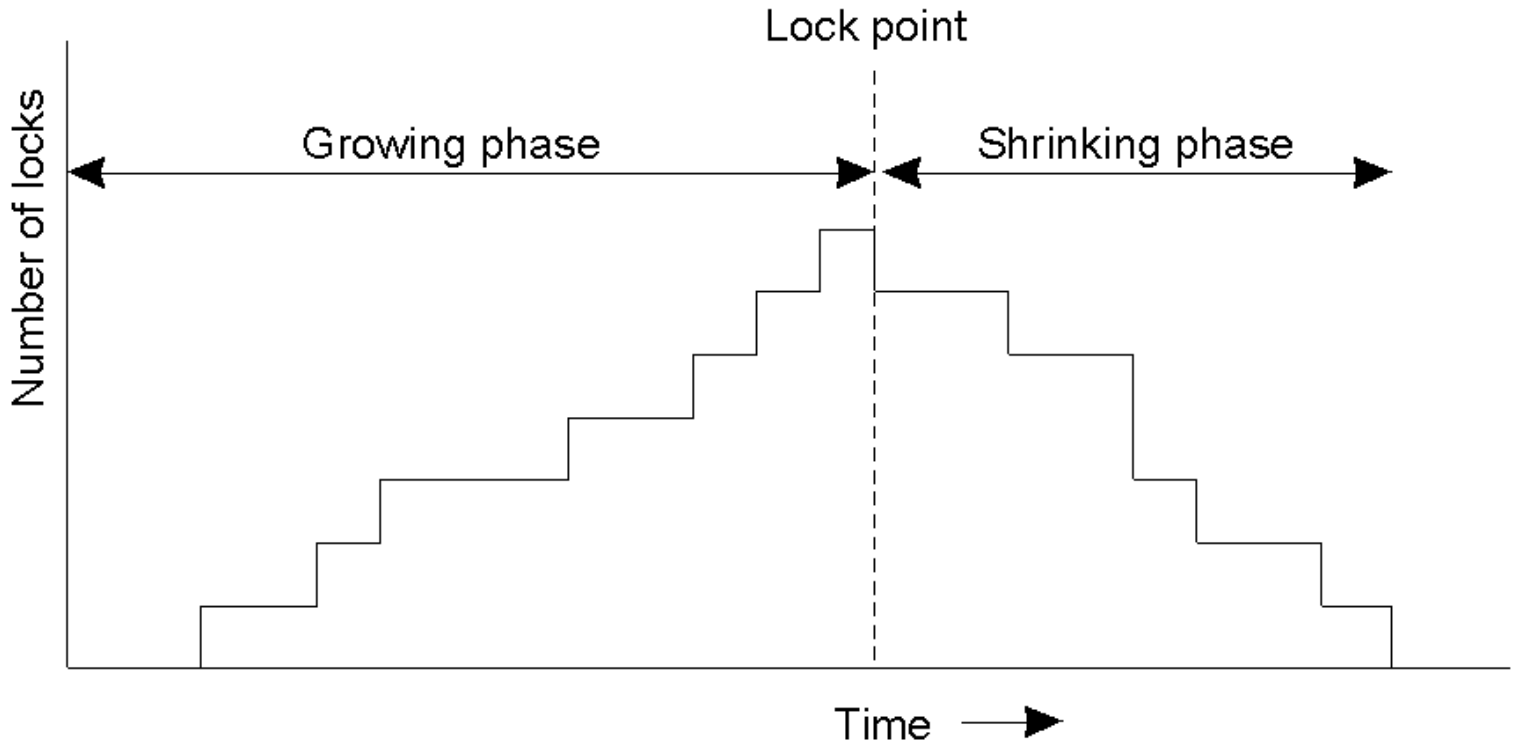
(c)

Schedule 1	x = 0; x = x + 1; x = 0; x = x + 2; x = 0; x = x + 3	Legal
Schedule 2	x = 0; x = 0; x = x + 1; x = x + 2; x = 0; x = x + 3;	Legal
Schedule 3	x = 0; x = 0; x = x + 1; x = 0; x = x + 2; x = x + 3;	Illegal

(d)

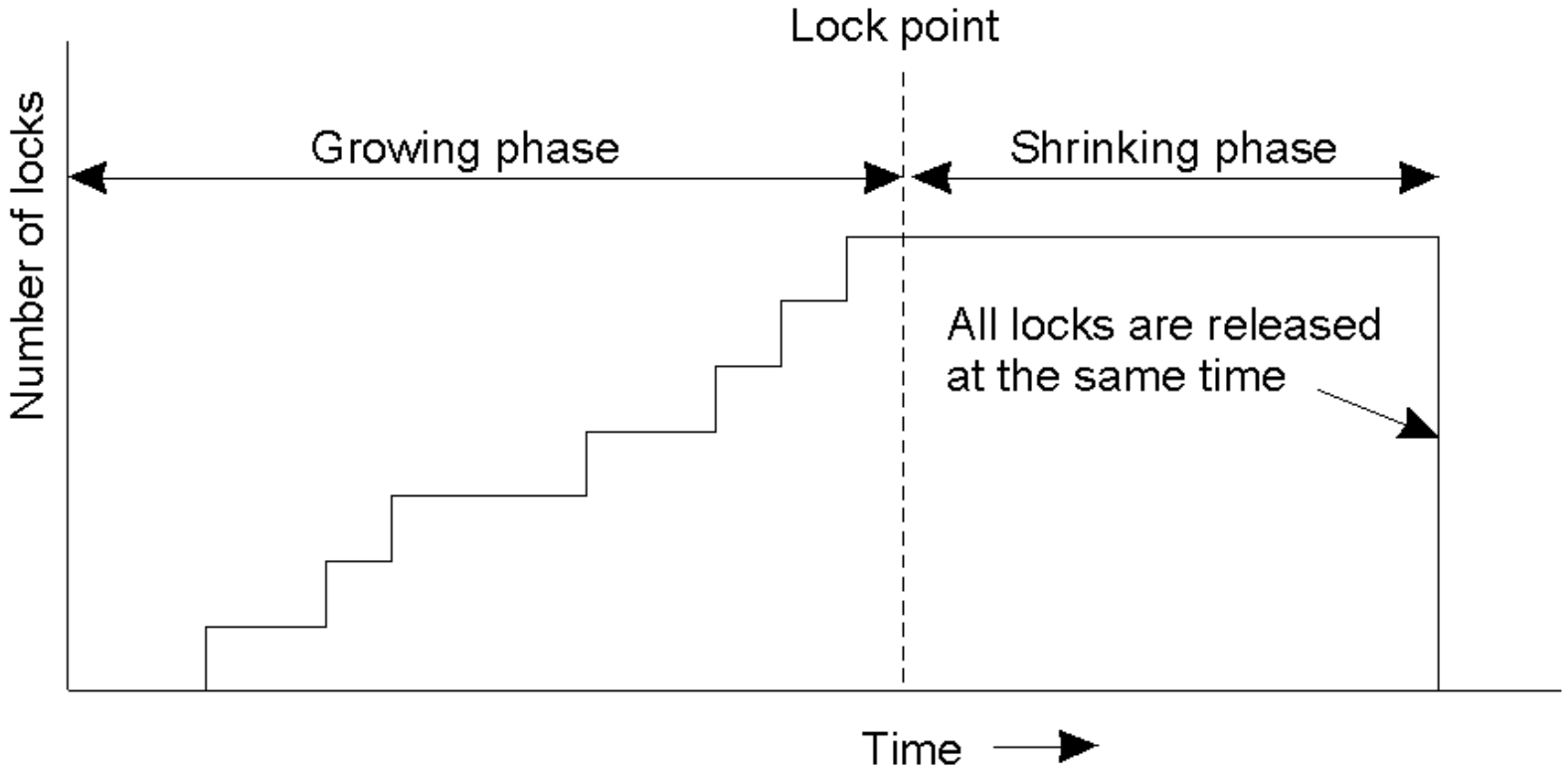
- a) – c) Three transactions T_1, T_2, and T_3
- d) Possible schedules

Two-Phase Locking (1)



- Two-phase locking.

Two-Phase Locking (2)



- Strict two-phase locking.

KG REDDY College of Engineering & Technology

Data integrity

Backup and recovery

- The problem of keeping a transaction atomic: successful or failed

  What if some of the intermediate steps failed?
- Log of database activity: use the log to undo a failed transaction.
- More problems: when to write the log, failure of the recovery system executing the log.

Security and access control

- Access rules for relations or attributes. Stored in a special relation (part of the data dictionary).
- Content-independent and content-dependent access control
- Content-dependent control: access to a view only or query modification

  (e.g. and-ing a predicate to the WHERE clause)
- Discretionary and mandatory access control

KG REDDY College of Engineering & Technology

Knowledge Bases and KBS (and area of AI)

- Information, Data, Knowledge (data in a form that allows reasoning)
- Basic components of a KBS
    - Knowledge base
    - Inference (reasoning) mechanism (e.g. forward/backward chaining)
    - Explanation mechanism/Interface
- Rule-based systems (medical diagnostics, credit evaluation etc.)

Rule Base:

1. IF (lecturing X)
   AND (grading-tests X)
   THEN (overworked X)

2. IF (month february)
   THEN (lecturing alison)

3. IF (month february)
   THEN (grading-tests alison)

4. IF (overworked X)
   THEN (bad-mood X)

5. IF (slept-badly X)
   THEN (bad-mood X)

6. IF (month february)
   THEN (weather cold)

7. IF (year 1993)
   THEN (economy bad)

Backward chaining:

Given the facts:

(month february)
(year 1993)

we can prove

(bad-mood alison)

Forward chaining:

Given the facts:

(month february)
(year 1993)

we can infer

(lecturing alison)
(grading-tests alison)
(overworked alison)
(bad-mood alison)
(economy bad)

Backward chaining by asking questions:

Given no facts we can try to prove
(bad-mood alison)

Then the system asks:
Is (month february) true?

If the answer is "yes", then the system proves successfully answers yes the initial question.
If the answer is "no" the system tries to find an alternative prove and asks:

Is (slept-badly alison) true ?