JavaEE平台技术

MySQL关系数据库原理

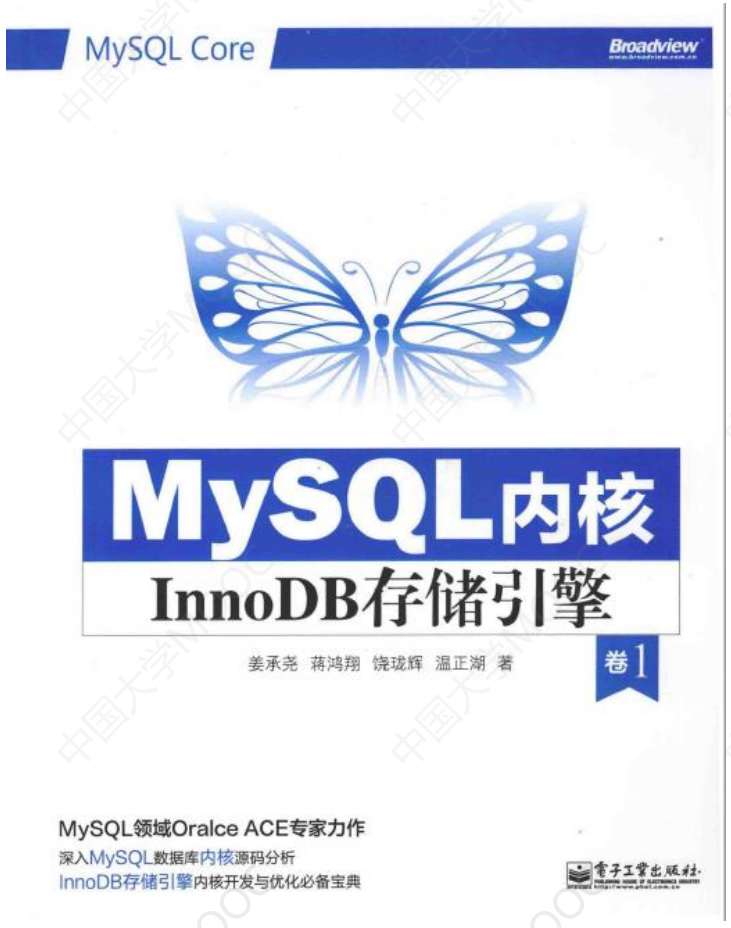
邱明 博士

厦门大学信息学院

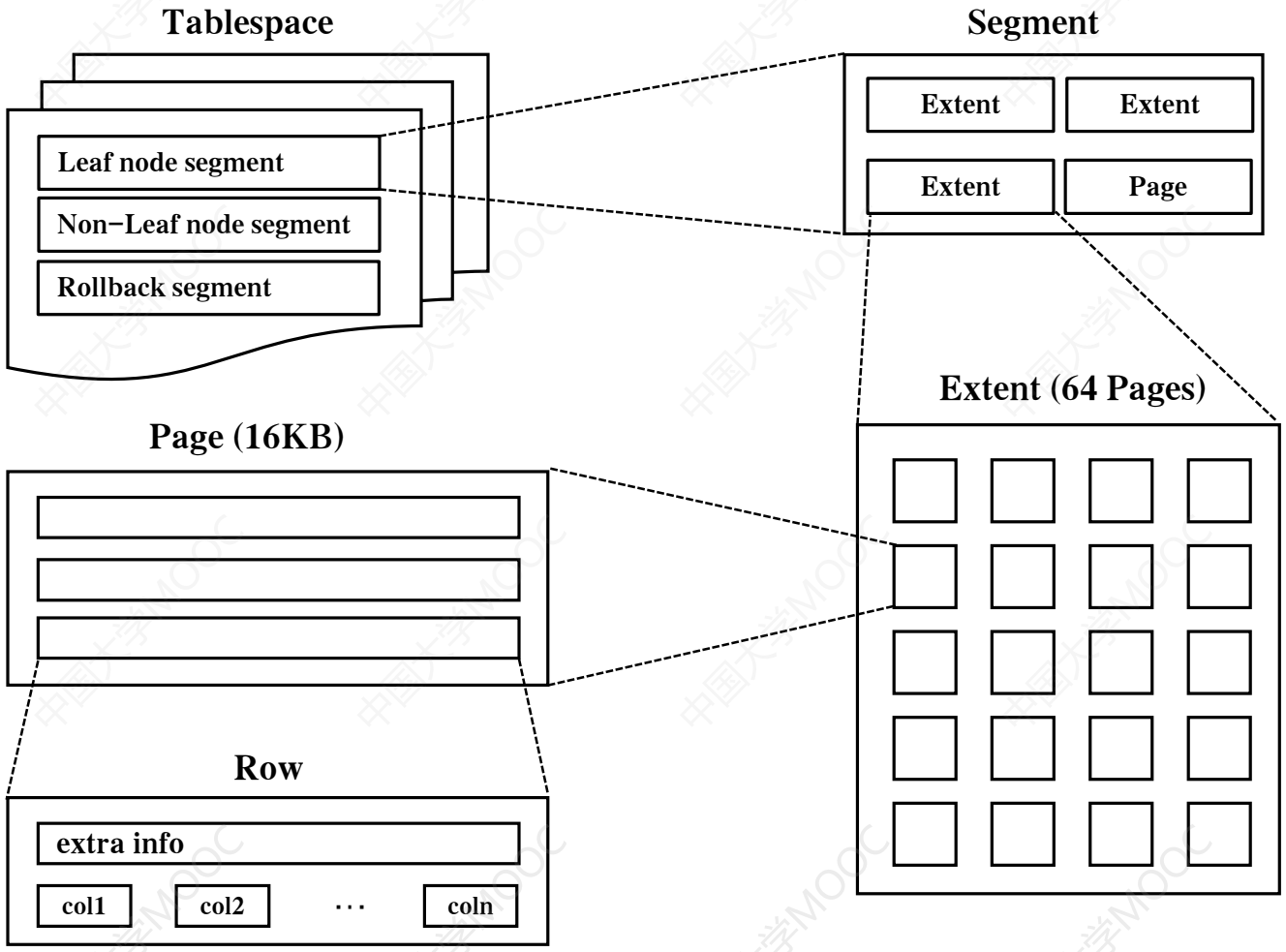
mingqiu@xmu.edu.cn

本章参考书

- MySQL内核：InnoDB存储引擎，第1卷
- 姜承尧等著
- 北京. 电子工业出版社, 2014.5
- ISBN 978-7-121-22908-4

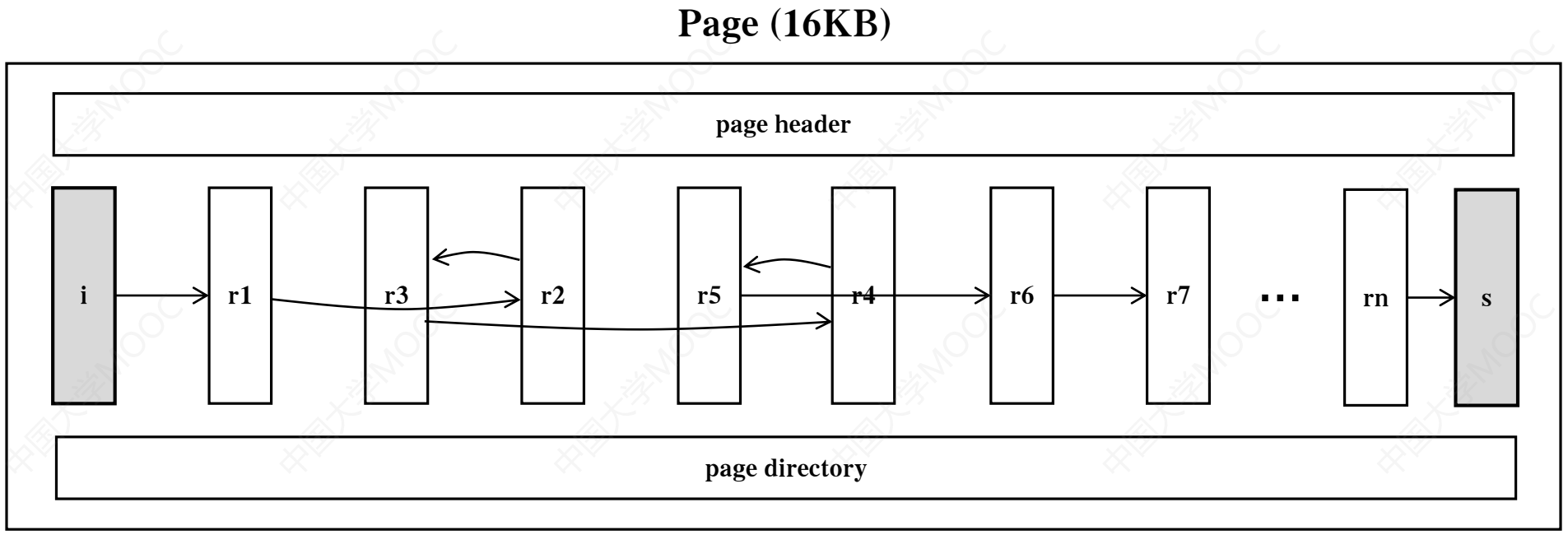


1. InnoDB的存储


Tablespace
Leaf node segment
Non−Leaf node segment
Rollback segment
Segment
Extent
Extent
Extent
Page
Extent (64 Pages)
Page (16KB)
Row
extra info
col1
col2
···
coln

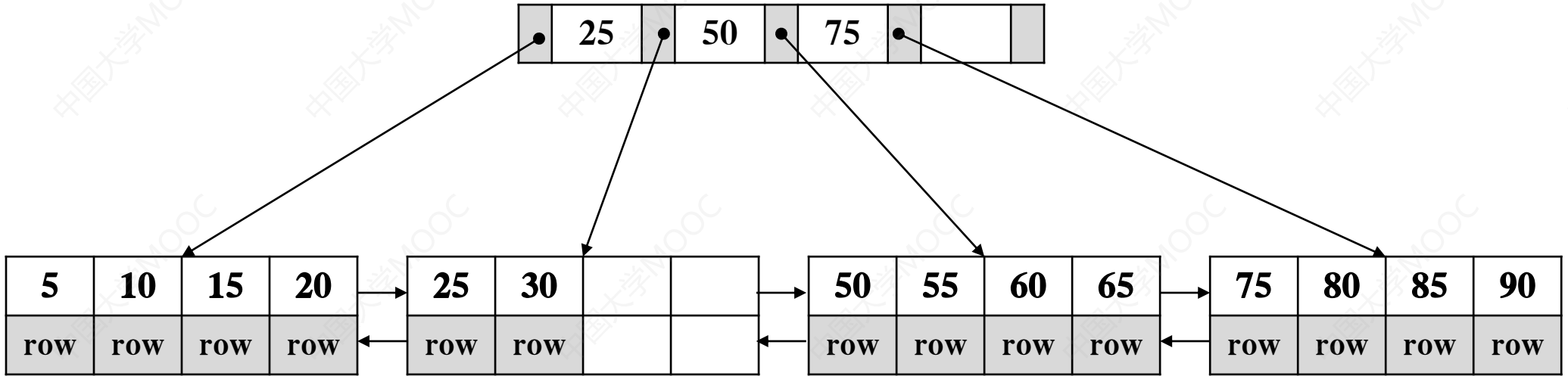
1. InnoDB的存储

- InnoDB的最小存储单位–页



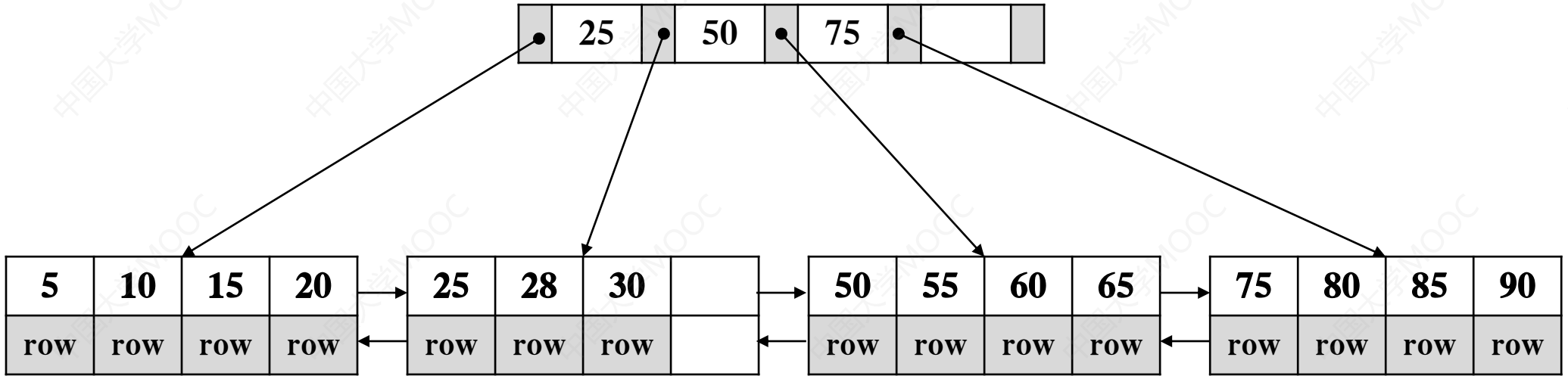
2. InnoDB的索引

- 索引的目的在于提高查询效率



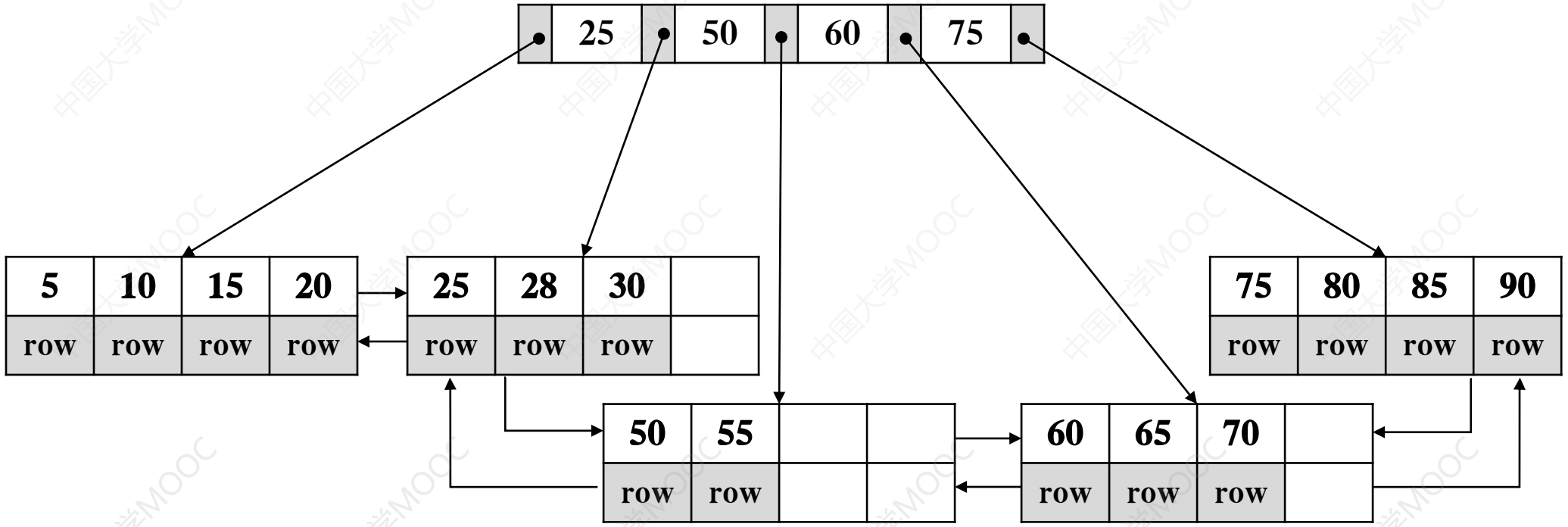
2. InnoDB的索引

- 插入（直接插入叶子节点）



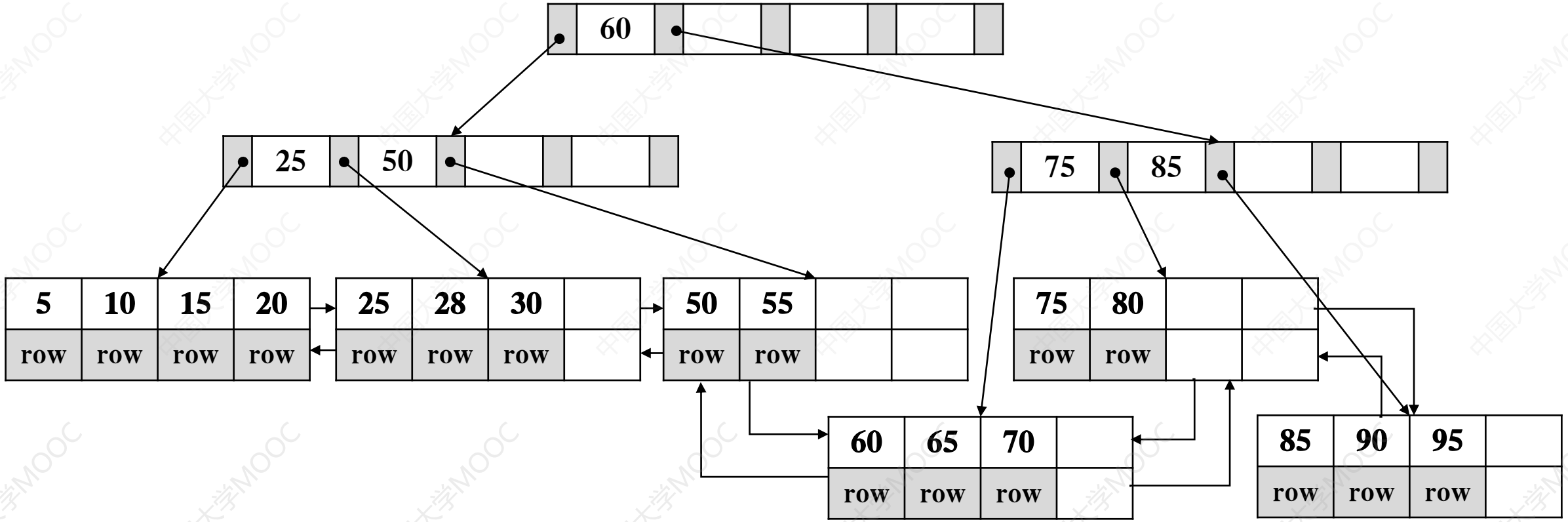
2. InnoDB的索引

- 插入（拆分叶子节点）



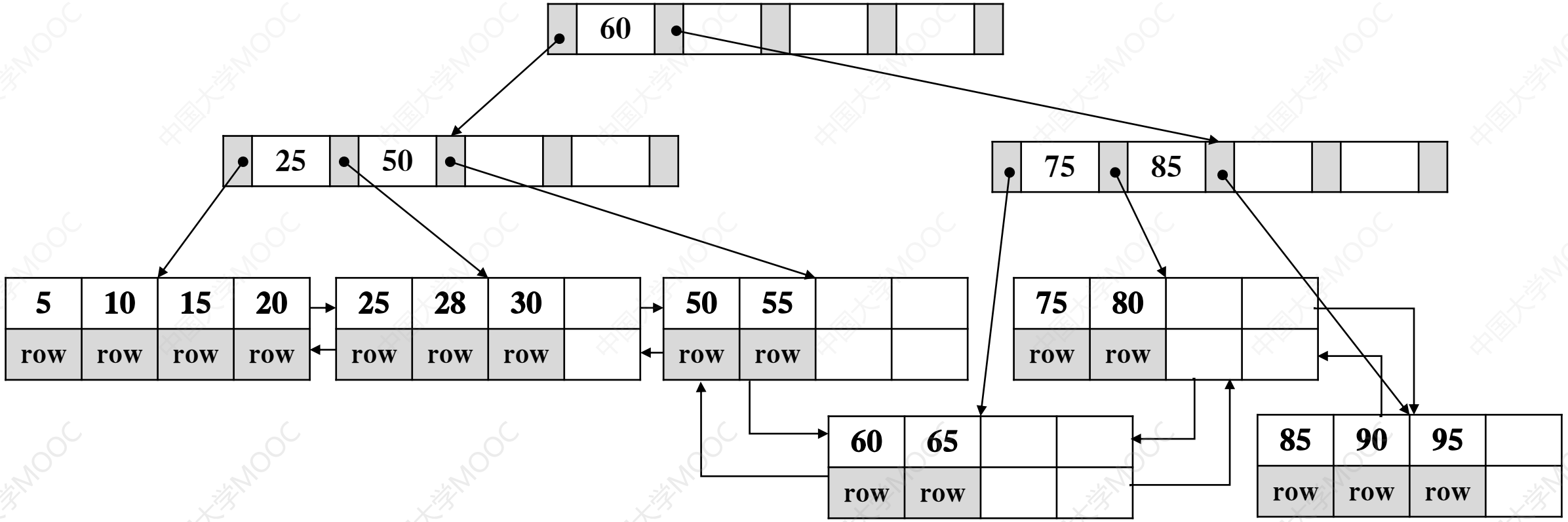
2. InnoDB的索引

- 插入（拆分中间节点和叶子节点）



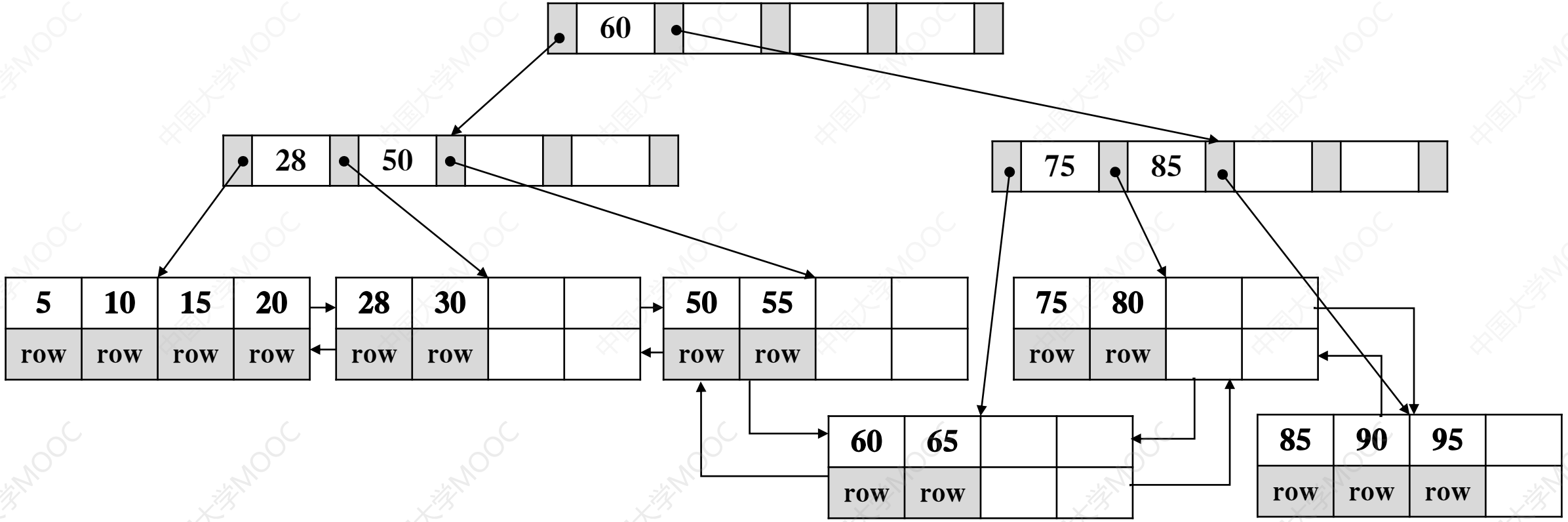
2. InnoDB的索引

- 删除（直接在叶子节点中删除70）



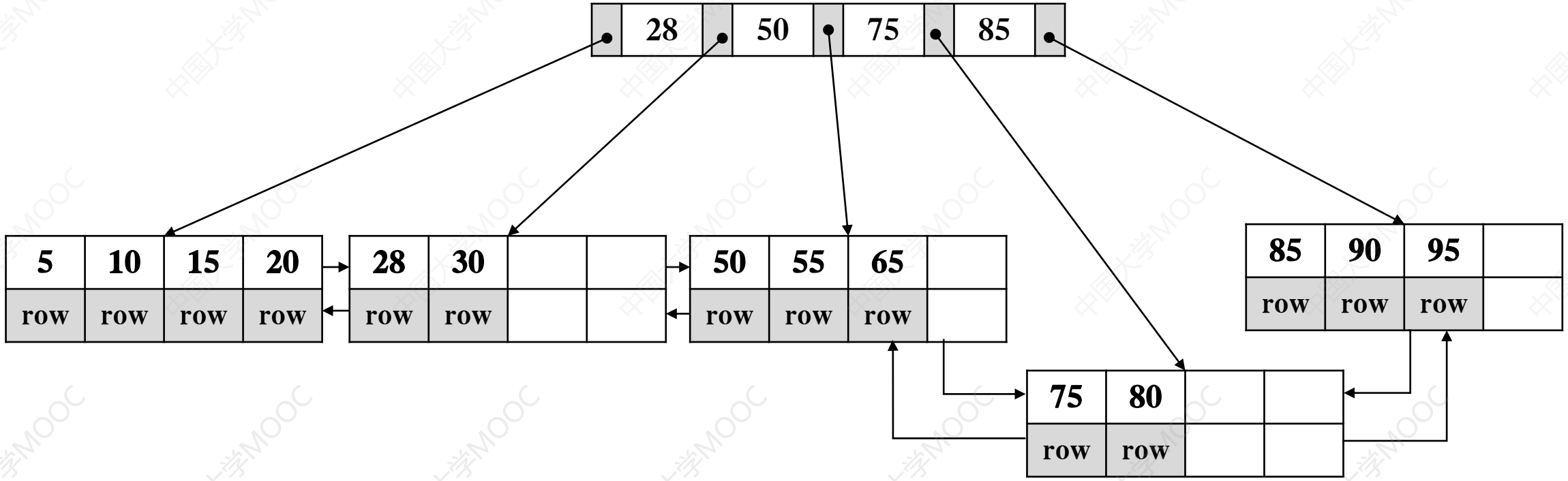
2. InnoDB的索引

- 删除（直接在叶子节点中删除25，将叶子节点中的右节点上移）



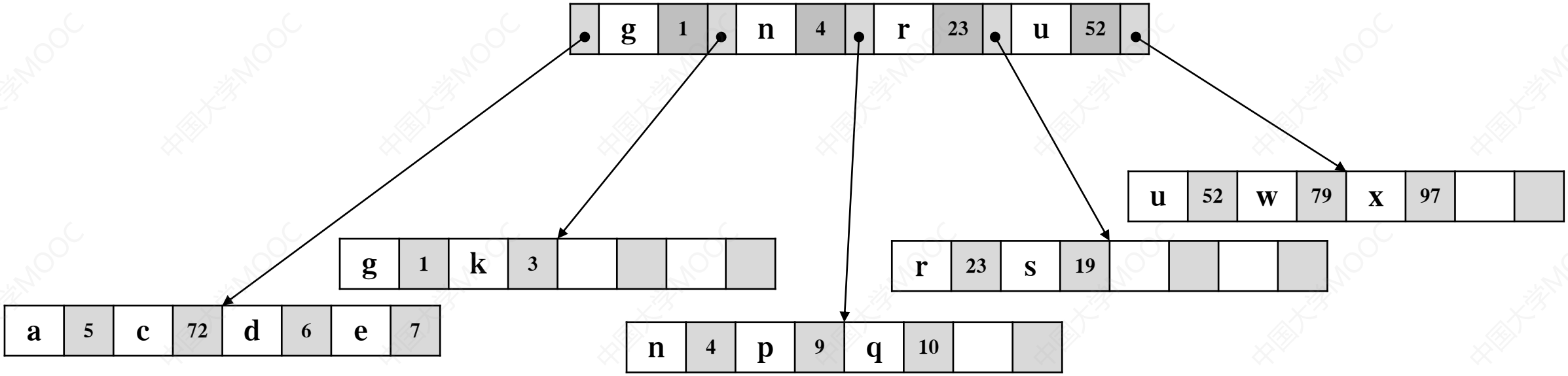
2. InnoDB的索引

- 删除（删除60）



2. InnoDB的索引

- 辅助索引



3. MySQL的事务

- ACID
  - Atomicity（原子性）：一个事务中的所有操作，要么全部完成，要么全部不完成，不会结束在中间某个环节。事务在执行过程中发生错误，会被回滚到事务开始前的状态。
  - Consistency（一致性）：在事务开始之前和事务结束以后，数据库的完整性约束没有被破坏。
  - Isolation（隔离性）：并发事务的操作对象之间相互隔离。
  - Durability（持久性）：事务一旦提交，对数据的修改就是永久的。

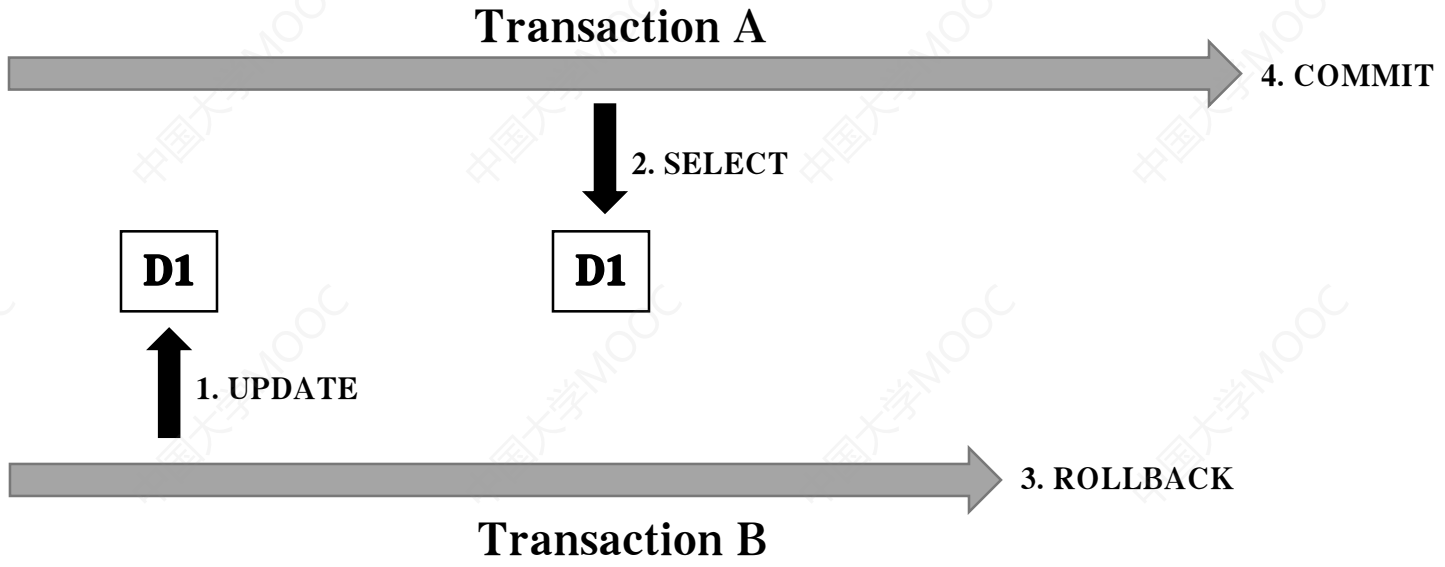
3. MySQL的事务

- 事务隔离级别（isolation）

事务隔离级别	脏读	不可重复读	幻读
READ UNCOMMITED	是	是	是
READ COMMITED	否	是	是
REPEATABLE READ	否	否	是
SERIALIZABLE	否	否	否

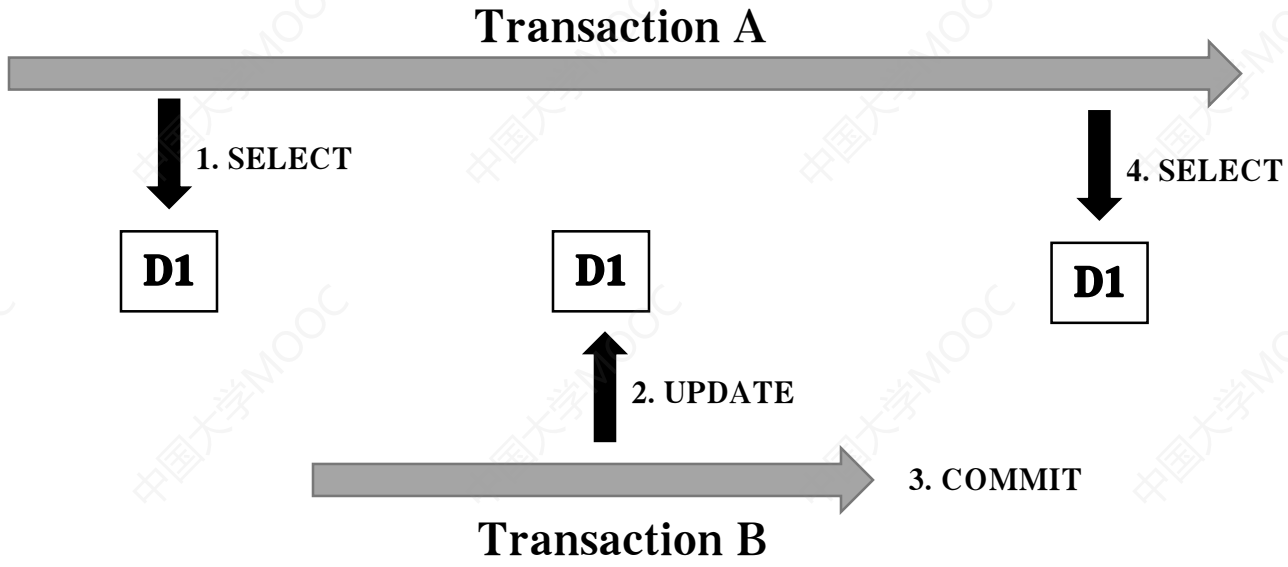
3. MySQL的事务

- READ UNCOMMITED – 脏读问题



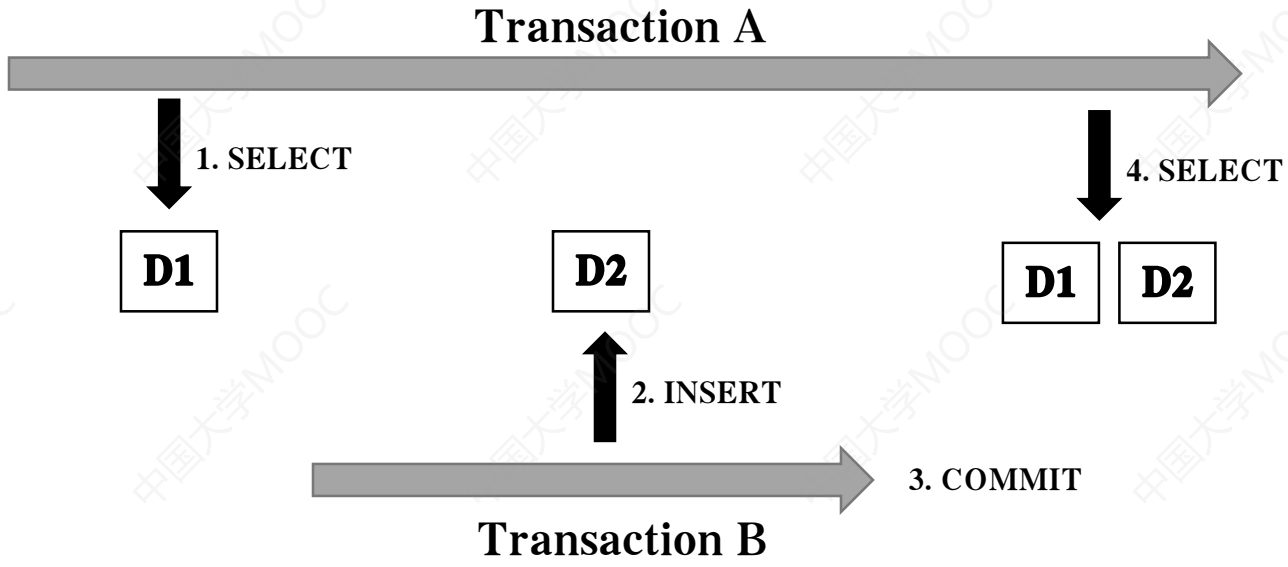
3. MySQL的事务

- READ COMMITED – 不可重复读



3. MySQL的事务

- REPEATABLE READ–幻读问题



3. MySQL的事务

- InnoDB引擎实现了两种锁
  - 共享锁（S Lock）-- 读锁
    - 允许多个事务对同一条记录加共享锁
  - 排他锁（X Lock）-- 写锁
    - 只允许一个事务对一条记录加锁

	X Lock	S Lock
X Lock	不兼容	不兼容
S Lock	不兼容	兼容

3. MySQL的事务

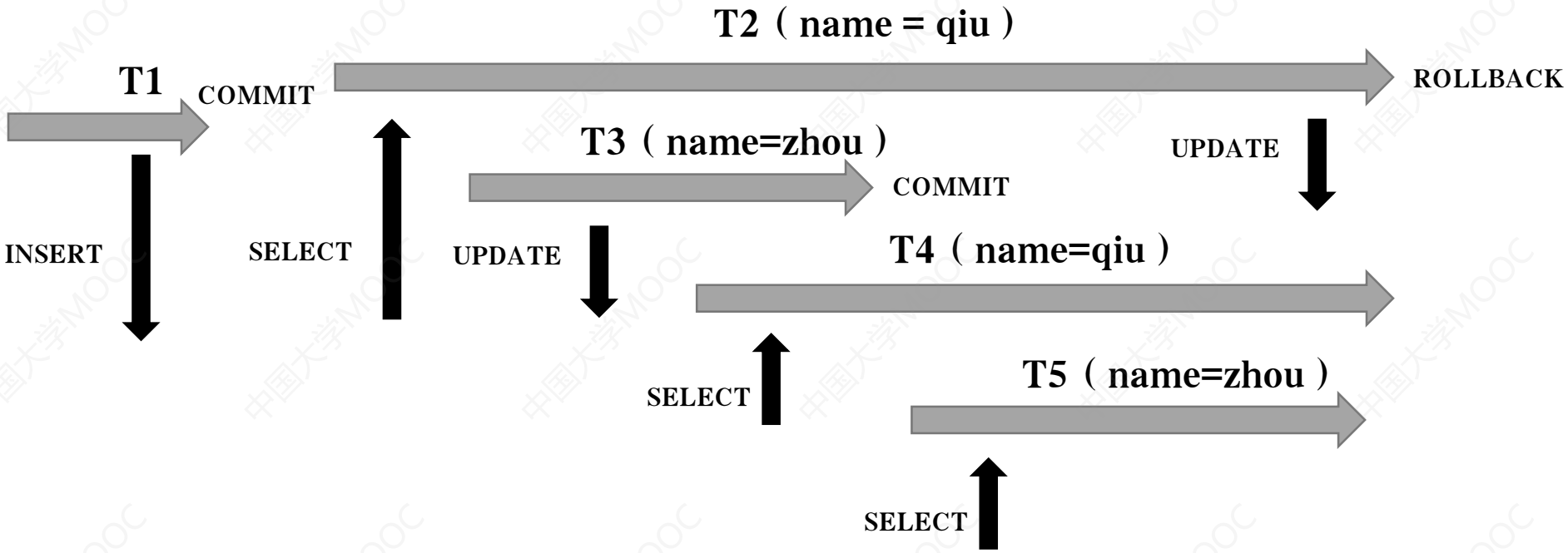
- 实现数据库的并发访问控制的最简单做法是
  - 读的时候不能写（允许多个线程同时读，即共享锁，S Lock）
  - 写的时候不能读（一次最多只能有一个线程对同一份数据进行写操作，即排他锁，X Lock）。

事务隔离级别	脏读	不可重复读	幻读	锁的方式
READ UNCOMMITED	是	是	是	无锁
READ COMMITED	否	是	是	对记录加写锁
REPEATABLE READ	否	否	是	对记录加读写锁
SERIALIZABLE	否	否	否	对表加读锁

4. MVCC

- MVCC：Multi-Version Concurrent Control 多版本并发控制
  - MVCC通过保存数据在某个时间点的快照来实现读的并发。这意味着一个事务无论运行多长时间，在同一个事务里能够看到数据一致的视图。根据事务开始的时间不同，同时也意味着在同一个时刻不同事务看到的相同表里的数据可能是不同的。
  - 每行数据都存在一个版本，每次数据更新时都更新该版本。
  - 修改时Copy出当前版本随意修改，各个事务之间无干扰。
  - 保存时比较版本号，如果成功（commit），则覆盖原记录；失败则放弃copy（rollback）

4. MVCC



id	name
1	qiu

id	name
1	zhou

4. MVCC

- Insert操作:
  - Transaction 1

insert into testmvcc values(1, “qiu");

Table: testmvcc

id	name	DB_TRX_ID	DB_ROLL_PTR
1	qiu	1	null

4. MVCC

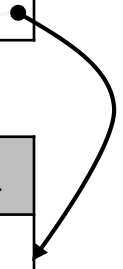
- Update
  - Transaction 2 **update table set name= ‘zhou' where id=1;**

Table: testmvcc

id	name	DB_TRX_ID	DB_ROLL_PTR
1	zhou	2	

Undo log

id	name	DB_TRX_ID	DB_ROLL_PTR
1	qiu	1	null



4. MVCC

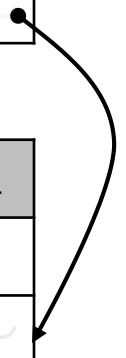
- Update
  - Transaction 3 **update table set name= ‘li' where id=1;**

Table: testmvcc

id	name	DB_TRX_ID	DB_ROLL_PTR
1	li	3	

Undo log

id	name	DB_TRX_ID	DB_ROLL_PTR
1	qiu	1	null
1	zhou	2	



4. MVCC

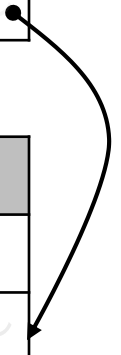
- Delete
  - Transaction 3 **delete from table where id=1;**

Table: testmvcc

	id	name	DB_TRX_ID	DB_ROLL_PTR
*	1	li	3	

Undo log

id	name	DB_TRX_ID	DB_ROLL_PTR
1	qiu	1	null
1	zhou	2	



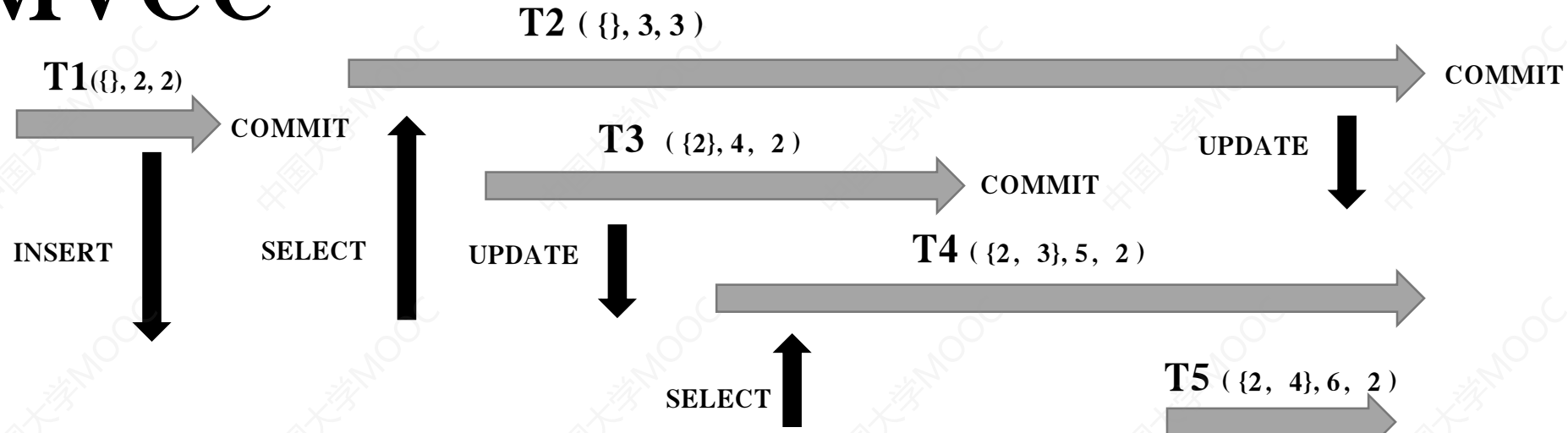
4. MVCC

- 查询操作--ReadView：
  - trx_ids: 当前系统中那些活跃(未提交)的读写事务ID。
  - low_limit_id: 下一个将被分配的事务ID。
  - up_limit_id: 活跃事务列表trx_ids中最小的事务ID，
    - 如果trx_ids为空，up_limit_id = low_limit_id。
  - creator_trx_id: 表示生成该 ReadView 的事务id

4. MVCC

- 查询规则：
  - 从最新的DB_TRX_ID开始依次判断下述条件，直到找到满足条件的数据
    - DB_TRX_ID = creator_trx_id，可见；（当前事务访问的是自己修改过的记录）
    - DB_TRX_ID >= low_limit_id，不可见；（记录在当前事务生成 ReadView 后才产生）
    - DB_TRX_ID < up_limit_id，可见；（记录在当前事务生成 ReadView 前已经提交）
    - (up_limit_id<=DB_TRX_ID<low_limit_id) and (DB_TRX_ID in trx_ids)，不可见；(创建记录的事务还未提交)
    - (up_limit_id<=DB_TRX_ID<low_limit_id) and NOT (DB_TRX_ID in trx_ids)，可见；(创建记录的事务已经提交)

4. MVCC



Undo log

id	name	DB_TRX_ID	DB_ROLL_PTR

testmvcc

id	name	DB_TRX_ID	DB_ROLL_PTR
3	qiu	1	null

Undo log

0x001

id	name	DB_TRX_ID	DB_ROLL_PTR
3	qiu	1	null

← HISTORY

testmvcc

id	name	DB_TRX_ID	DB_ROLL_PTR
3	zhou	3	0x001

Undo log

0x001

id	name	DB_TRX_ID	DB_ROLL_PTR
3	qiu	1	null

testmvcc

id	name	DB_TRX_ID	DB_ROLL_PTR
3	zhou	3	0x001

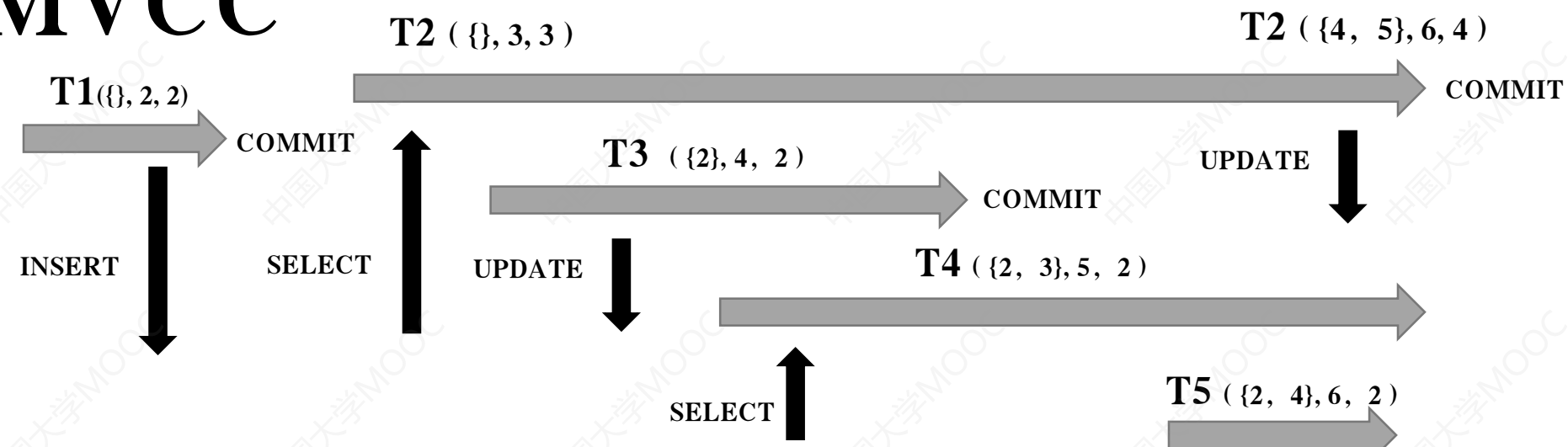
Undo log

	id	name	DB_TRX_ID	DB_ROLL_PTR	
0x001	3	qiu	1	null	← HISTORY
0x002	3	zhou	3	0x001	← HISTORY

testmvcc

id	name	DB_TRX_ID	DB_ROLL_PTR
3	li	2	0x002

4. MVCC



Undo log

id	name	DB_TRX_ID	DB_ROLL_PTR

testmvcc

id	name	DB_TRX_ID	DB_ROLL_PTR
3	qiu	1	null

Undo log

0x001

id	name	DB_TRX_ID	DB_ROLL_PTR
3	qiu	1	null

← HISTORY

testmvcc

id	name	DB_TRX_ID	DB_ROLL_PTR
3	zhou	3	0x001

Undo log

0x001

id	name	DB_TRX_ID	DB_ROLL_PTR
3	qiu	1	null

testmvcc

id	name	DB_TRX_ID	DB_ROLL_PTR
3	zhou	3	0x001

Undo log

0x001

id	name	DB_TRX_ID	DB_ROLL_PTR
3	qiu	1	null

← HISTORY

0x002

id	name	DB_TRX_ID	DB_ROLL_PTR
3	zhou	3	0x001

testmvcc

id	name	DB_TRX_ID	DB_ROLL_PTR
3	li	3	0x002

4. MVCC

- READ COMMITED – 每次读的时候产生ReadView
- REPEATABLE READ – 事务开始的时候产生ReadView