3-Level Example

Insert Into Persons
Values (Name=..., City="Austin", Age=29, ...)

Select Name
From Persons
Where City="Seattle"
And Age=29

Select Name
From Persons
Where Age=30

store(x)

insert (CityIndex, "Austin", @x)

search (CityIndex, "Seattle")

insert (AgeIndex, 29, @x)

search (AgeIndex, 29)

fetch(y)

r(p)
w(p)
r(r)
r(n)

r(r) r(l) r(n) w(l) r(l) r(r"r(n"r(l")r(r"r(n"r(l")w(l")r(p)w(p)

r(r"r(n"r(l") r(p) w(p)

r(r"r(n"r(l") r(p) w(p)
Protocols

- FOCC at the page level, strong 2PL at the record level
- FOCC at the page level, strong 2PL at the query level
- ROMV at the page level, strong 2PL at the record level
- ROMV at the page level, strong 2PL at the query level
Predicate-based locking

- locks are related to predicates from the query (WHERE...)
- locks are intensional
- there is a conflict between two queries (r-w), if there are tuples that turn the conjunction of predicates into TRUE
- computationally expensive (SAT problem)
- only limited (but very important) cases are of practical interest. See index structures
Two types of transactions:

- Update: lock-based, S2PL. Gets a timestamp at the END
- Read-only: reads the most recent committed version. Gets a timestamp at the START
Transaction chopping

- Split transactions into sub-transactions that run (potentially) concurrently
- Chopping is correct if every execution of the transaction pieces is conflict equivalent to some serial history of original transactions
- Rule 1: precedence from the original schedule is kept
- Rule 2: each piece is executed according to some CC protocol
Transaction chopping

- \( t_1 = r_1(x)w_1(x)r_1(y)w_1(y) \)
- \( t_2 = r_2(x) \)
- \( t_3 = r_3(y)w_3(y) \)
- \( t_{11} = r_1(x), \ t_{12} = w_1(x), \ t_{13} = r_1(y)w_1(y) \)
Homework: Task 1

- Given is the B⁺-tree index on AccountNumber (see next slide)
- All nodes have capacity of 4
- Which locks need to be requested for the following transaction
  ```sql
  begin transaction;
  select Count(*) From Accounts
  where AccountNumber between 11 and 25;
  insert into Accounts (AccountNumber,...) values (27,...);
  commit transaction;
  ```
- Assume incremental key range locking at the access layer and lock coupling at the page layer
Task 1: Index
Task 2:

- Give examples of schedules that fall into the following five isolation-level classes (but not into the next larger, surrounding class):
  - not even read uncommitted
  - read uncommitted
  - read committed
  - repeatable read
  - serializability.
Homework due Feb 3: Part 1

\[ s_1 = w_1(x)w_1(y)r_2(u)w_2(x)r_2(y)w_2(y)a_2w_1(z)c_1 \] and
\[ s_2 = w_1(x)w_1(y)r_2(u)w_2(x)r_2(y)w_2(y)w_1(z)a_1c_2 \]
Determine \( \exp(s_1) \) and \( \exp(s_2) \) as well as the corresponding reductions.

Which of the properties RC, ST, RG, PRED, and LRC are satisfied by the following schedules:

\[ s_1 = r_1(a)r_2(a)w_1(a)c_1c_2 \]
\[ s_2 = r_1(a)w_1(a)r_2(b)w_2(b)w_2(a)c_2c_1 \]
\[ s_3 = r_1(a)\text{incr}_2(a)\text{incr}_2(b)\text{incr}_3(b)c_3a_2c_1 \]

Some of the above properties cannot be applied to a \( s_3 \). Try to define generalizations of these properties to flat object schedules with arbitrary operations.
Info

- Exercises due: 9 AM, January 27, 2014 (except for those marked due Feb 3)
- Submit to andrey.gubichev@in.tum.de