



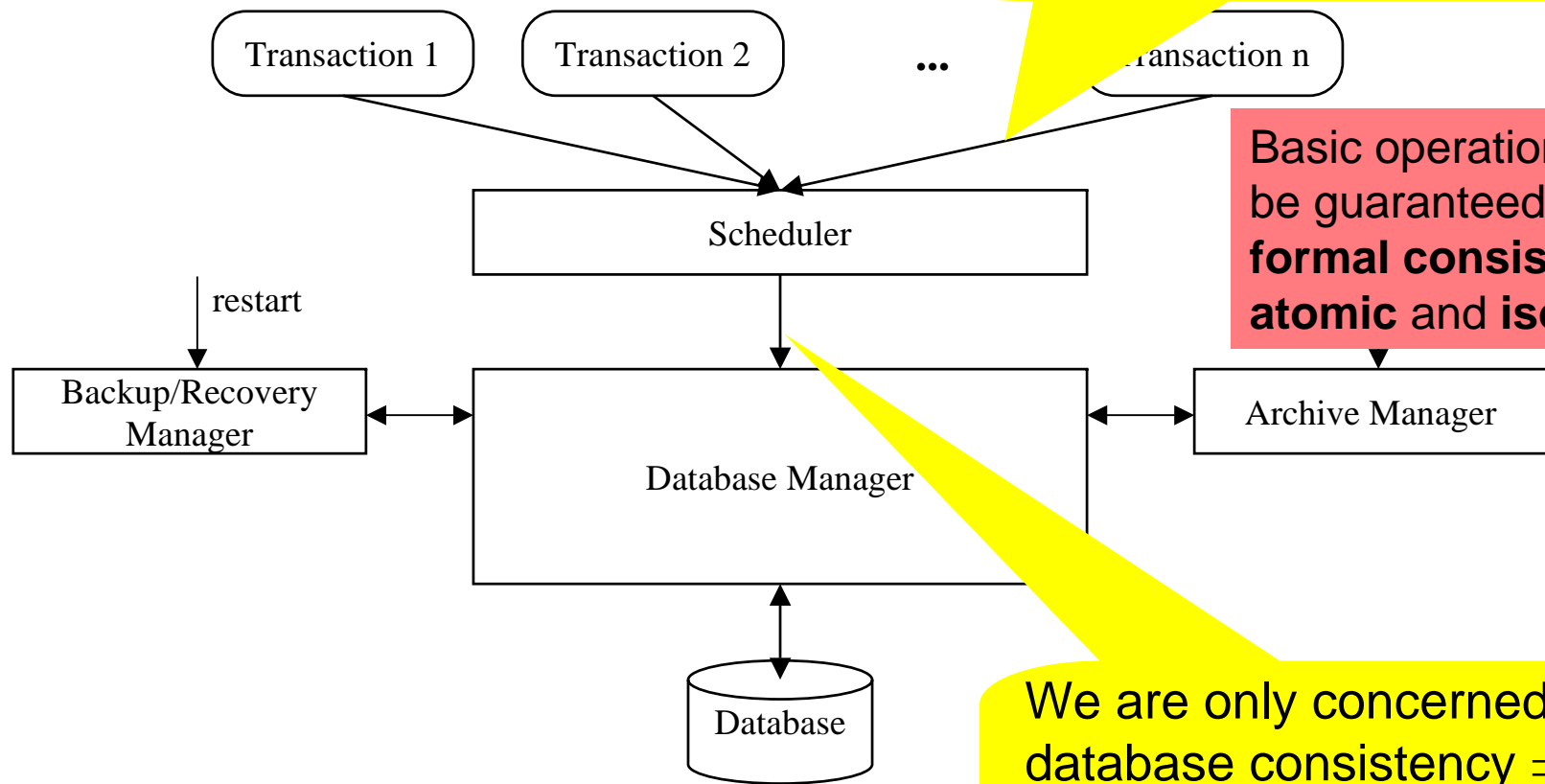
Chapter 2

Model for transactions

Read/write model

Basic operations in the model?

We abstract from transaction semantics except for database reads and writes.



Basic operations must be guaranteed to be **formal consistent, atomic and isolated.**

We are only concerned with database consistency \Rightarrow we are only interested in database reads and writes.

Read/write model

“Syntax”:

Definition 2.1 (Read/Write Model Transaction):

A **transaction** t is a partial order of steps (actions) of the form $r(x)$ or $w(x)$, where $x \in D$ (D database) and **reads and writes** applied to the same object are strictly ordered.

We write $t = (op, <)$

for transaction t with step set op and partial order $<$.

Examples: $t_1 = r(s) \rightarrow w(s) \rightarrow r(v) \rightarrow w(v)$

for short: $r(s) \ w(s) \ r(v) \ w(v)$

$t_2 = r(s) \rightarrow w(s)$
 $r(v) \rightarrow w(v)$

Read/write model

“Syntax”:

Definition 2.1 (Read/Write Model Transaction):

A **transaction** t is a partial order of steps (actions) of the form $r(x)$ or $w(x)$, where $x \in D$ (D database) and reads and writes applied to the same object are strictly ordered.

We write $t = (op, <)$

for transaction t with step set op and partial order $<$.

“Semantics”:

Interpretation of j^{th} step, p_j , of t :

If $p_j=r(x)$, then interpretation is assignment $v_j := x$ to local variable v_j

If $p_j=w(x)$ then interpretation is assignment $x := f_j(v_{j_1}, \dots, v_{j_k})$.

with unknown function f_j and j_1, \dots, j_k denoting t 's prior read steps.

Worst-case assumption!

Read/write model

Examples: $t_1 = r(s) \rightarrow w(s) \rightarrow r(v) \rightarrow w(v)$
for short: $r(s) w(s) r(v) w(v)$

$t_2 = r(s) \rightarrow w(s)$
 $r(v) \rightarrow w(v)$

“Semantics”:

Interpretation of j^{th} step, p_j , of t :

If $p_j = r(x)$, then interpretation is assignment $v_j := x$ to local variable v_j

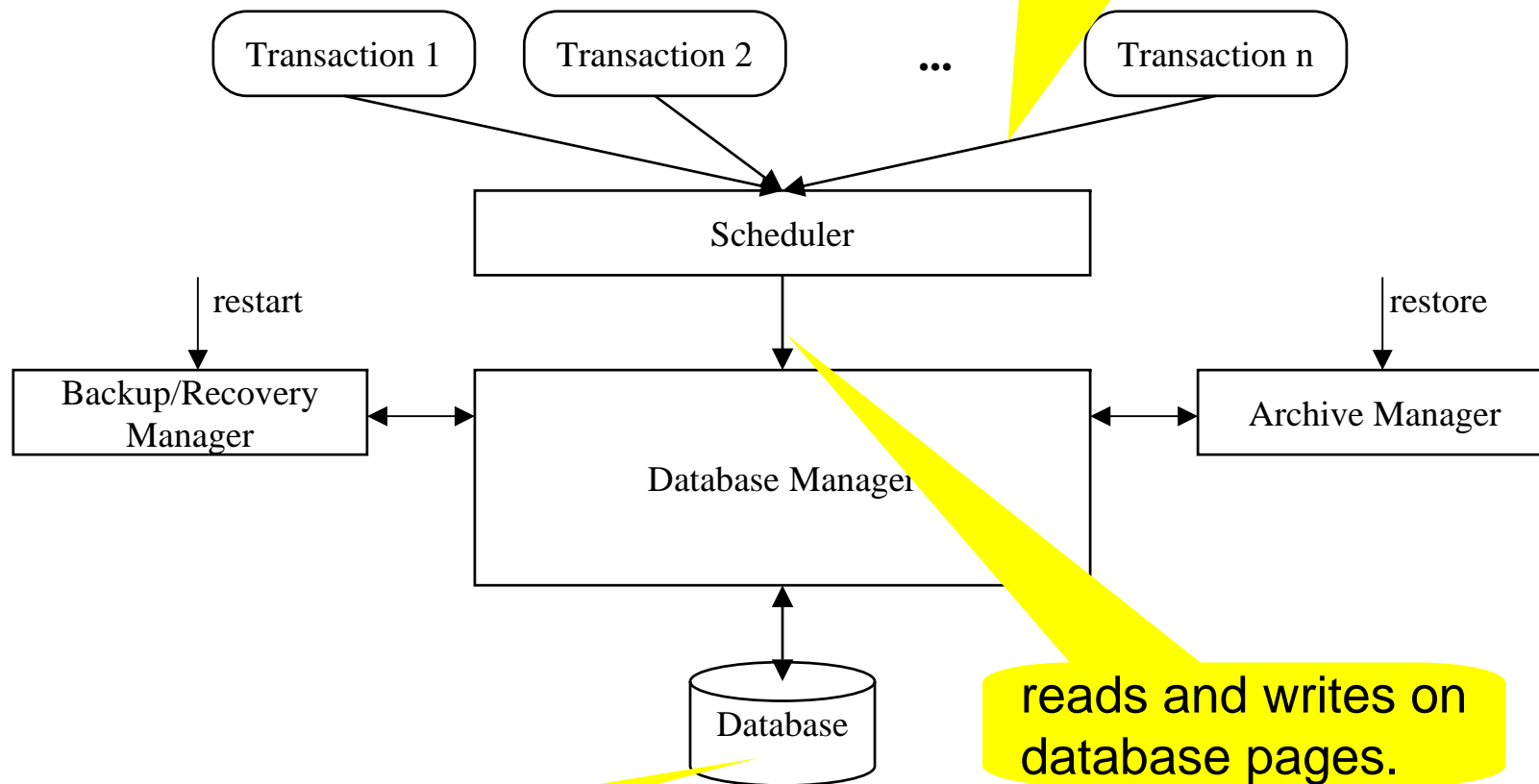
If $p_j = w(x)$ then interpretation is assignment $x := f_j(v_{j_1}, \dots, v_{j_k})$.

with unknown function f_j and j_1, \dots, j_k denoting p_j 's prior read steps.

Page model

Basic data elements in the model?

reads and writes on database pages.

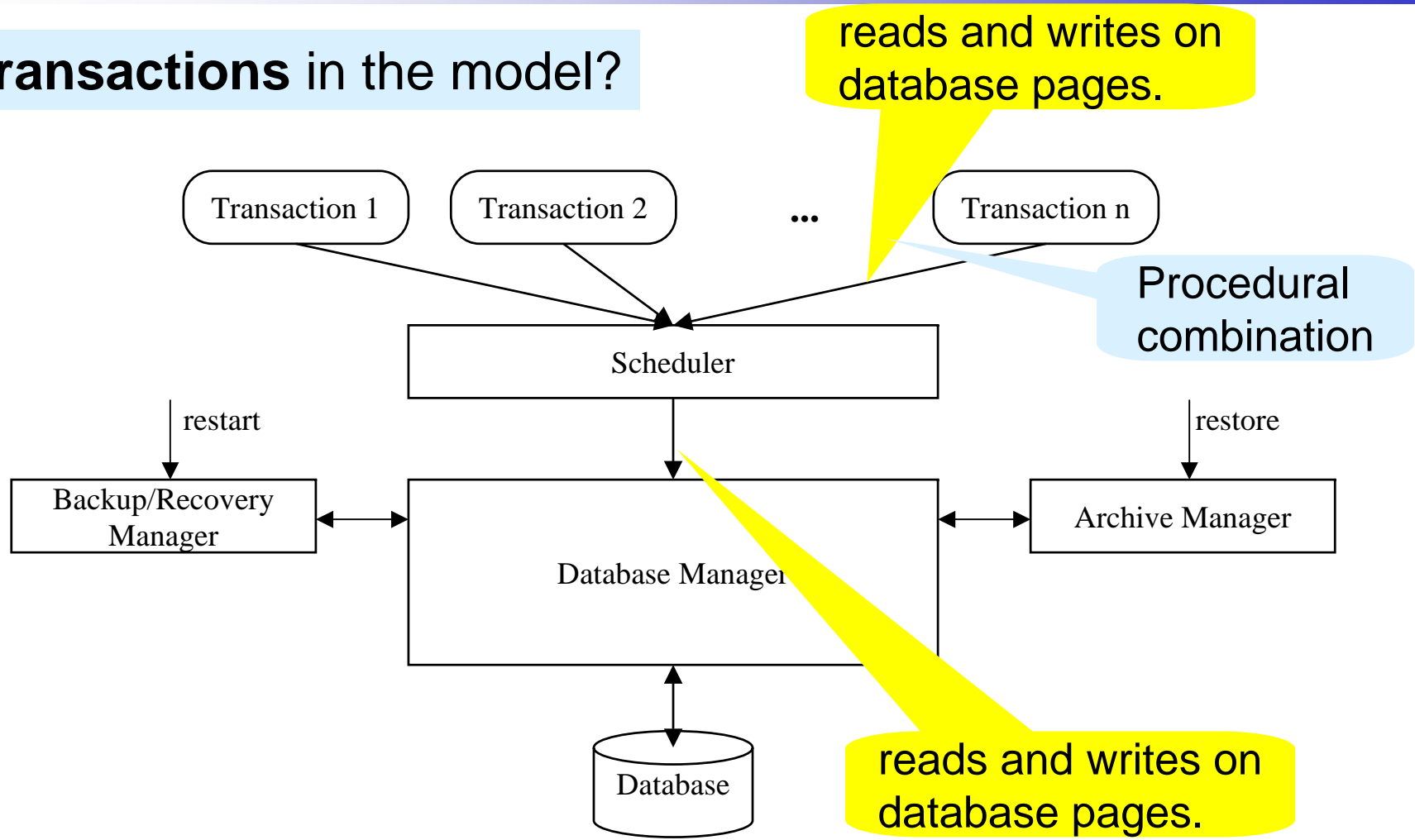


reads and writes on database pages.

Organized in data pages.

Procedural model

Transactions in the model?



Procedural model

“Syntax”:

Definition 2.1 (Read/Write Model Transaction):

A **transaction** t is a **partial order** of steps (actions) of the form $r(x)$ or $w(x)$, where $x \in D$ (D database) and reads and writes applied to the same object are strictly ordered.

We write $t = (op, <)$

for transaction t with step set op and **partial order** $<$.

Examples: $r(s) \rightarrow w(s)$
 $r(v) \rightarrow w(v)$

- For some operations their mutual order of execution does not affect the end result.
 - Definition covers the multiprocessor/parallel case.

Procedural model

A transaction brackets a number of operations

■ read

r

■ write

w

by

abbrev.

■ begin transaction and

b

■ end transaction or

c

■ abort

a

Successful completion of the transaction.
Results are made persistent (**commit**).

Unsuccessful termination of the transaction.
Results are discarded.

Transaction model

Definition 2.2 (Read/Write Model Transaction):

A **transaction** t_i is a partial order $(op_i, <_i)$, with op_i all operations in t_i and

$$1. op_i \subseteq \{r_i(x), w_i(x) \mid x \text{ is data element}\} \cup \{a_i, c_i\}$$

$$(op'_i = op_i \setminus \{a_i, c_i\})$$

$$2. \forall p \in op_i \ p = c_i \vee p = a_i \succ (\forall q \in op'_i \ q <_i p)$$

$$3. r_i(x), w_i(x) \in op_i \succ (r_i(x) <_i w_i(x) \vee w_i(x) <_i r_i(x))$$

$$4. a_i \in op_i \succ c_i \notin op_i \text{ and } c_i \in op_i \succ a_i \notin op_i$$

Each operation occurs at most once

If a or c in t_i , then as the last operation

Reads and writes on the same element are ordered

If a or c in t_i , only one of them

Transaction model

Definition 2.2 (Read/Write Model Transaction):

A **transaction** t_i is a partial order $(op_i, <_i)$, with op_i all operations in t_i and

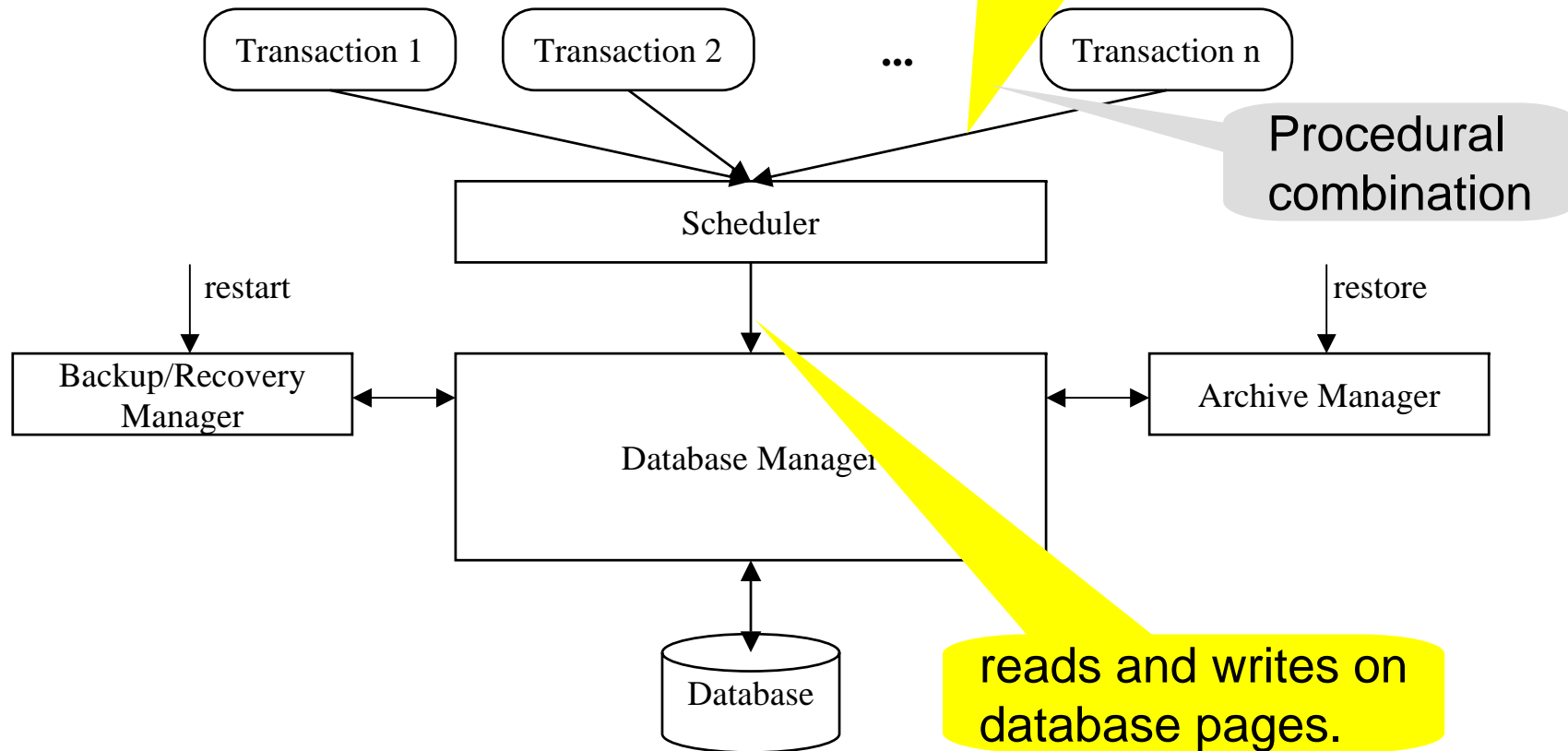
1. $op_i \subseteq \{r_i(x), w_i(x) \mid x \text{ is data element}\} \cup \{a_i, c_i\}$
($op'_i = op_i \setminus \{a_i, c_i\}$)
2. $\forall p \in op_i \ p = c_i \vee p = a_i \succ (\forall q \in op'_i \ q <_i p)$
3. $r_i(x), w_i(x) \in op'_i \succ (r_i(x) <_i w_i(x) \vee w_i(x) <_i r_i(x))$
4. $a_i \in op_i \succ c_i \notin op_i$ and $c_i \in op_i \succ a_i \notin op_i$

Often excluded due to standard implementation:
No read of an element after it was updated.

Transaction set model

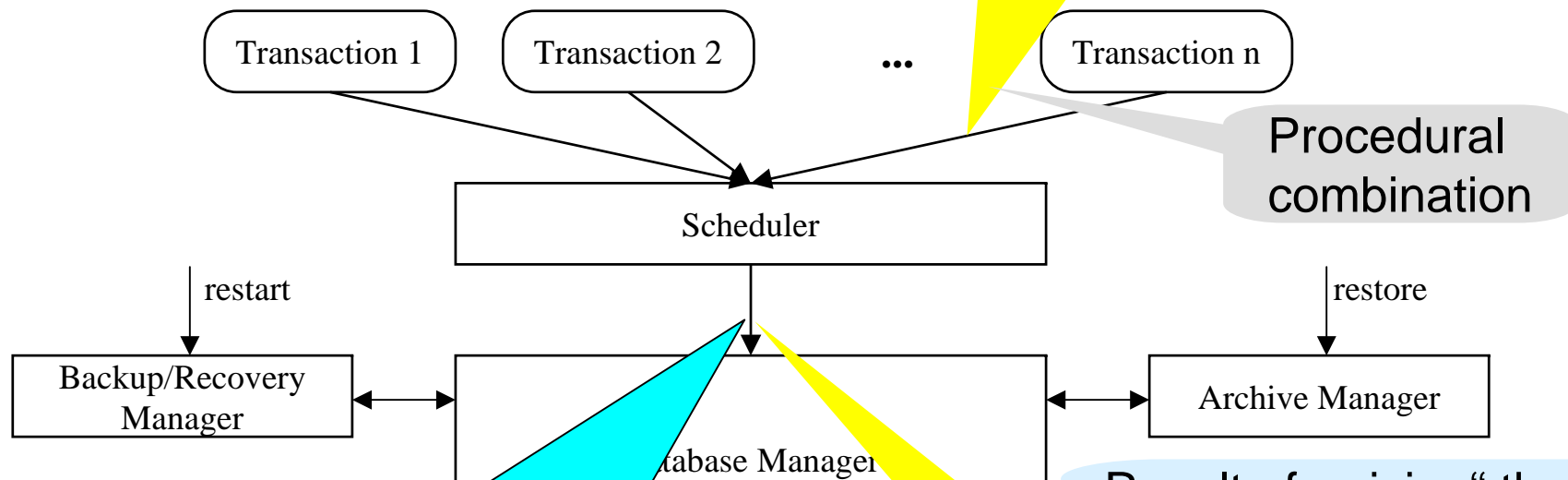
Transaction sets in the model?

reads and writes on database pages.



Transaction set model

Notation: $T = \{t_1, \dots, t_n\}$ is the set of transactions covered by a schedule.



Result: Schedule.

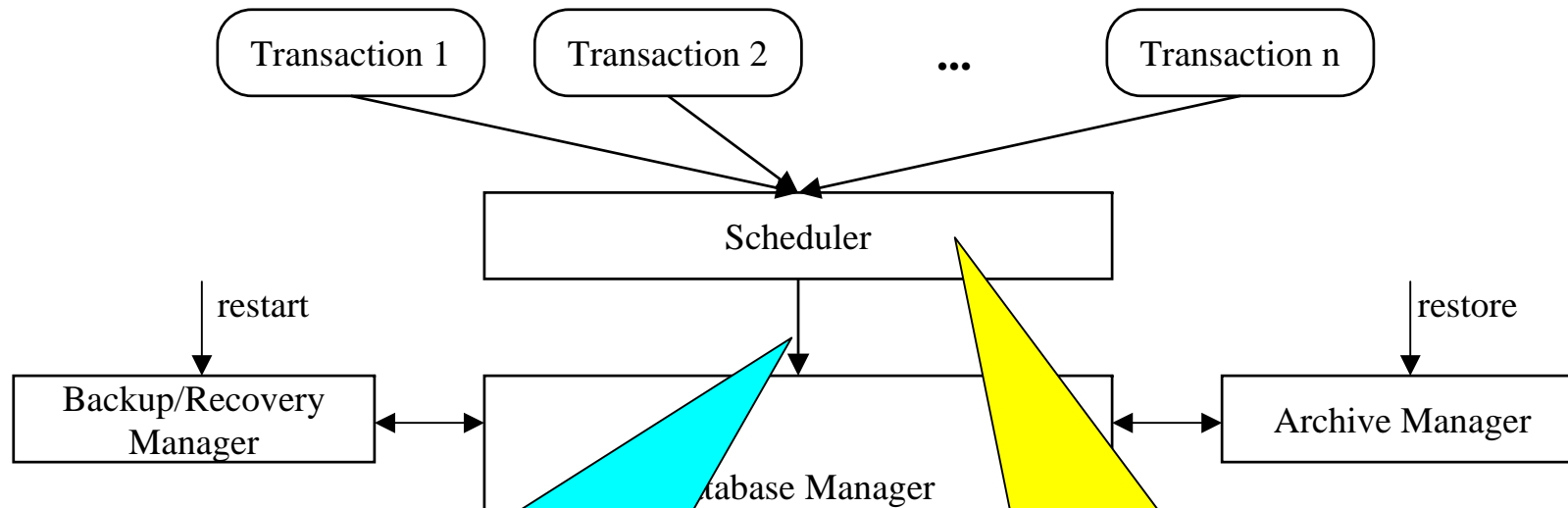
- Not every syntactically correct schedule is semantically correct (i.e., preserves the ACID properties).
- Formal description of correct schedules needed.

Result of „mixing“ the of different transactions?

reads and writes on database pages.

Transaction set model

Algorithms for the runtime generation of semantically correct schedules?



Result: **Schedule**.

- Not every syntactically correct schedule is semantically correct (i.e., preserves the ACID properties).
- Formal description of correct schedules needed.

- Application programs send basic operations.
- Algorithm sorts these dynamically into a correct schedule.