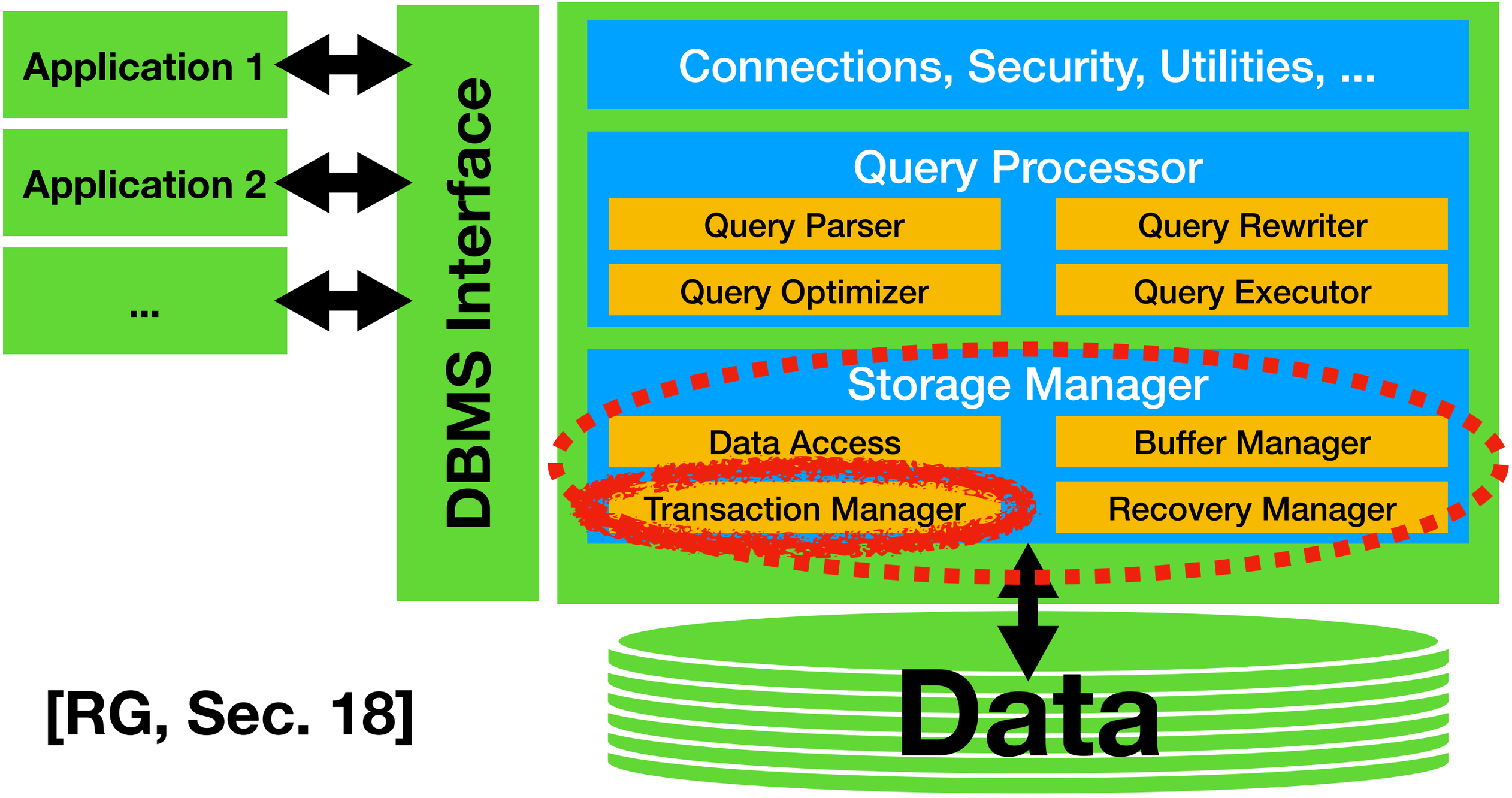
# Transactions

Immanuel Trummer
itrummer@cornell.edu
www.itrummer.org

# Database Management Systems (DBMS)




[RG, Sec. 18]

# Database Transactions

- A sequence of updates (or queries) that is "**connected**"
- **DBMS commands** for assigning queries to transactions
- DBMS makes certain **guarantees** about transactions
  - E.g., executes **entire** transaction **or no part** of it

# Example Transaction

- Want to **wire $50** from Alice to Bob
  - Step 1: **Subtract** $50 from Alice's account
  - Step 2: **Add** $50 to Bob's account
- Both steps are semantically **related**
  - I.e., want to execute **both or none**

# Transactions in Postgres

- Begin a transaction with command **BEGIN;**
- End a transaction with command **COMMIT;**
- Everything in between belongs to **transaction**

# ACID Guarantees

- Most RDBMS give **ACID guarantees** for transactions
- **A: Atomicity** (either execute all or nothing)
- **C: Consistency** (enforce all integrity constraints)
- **I: Isolation** (avoid interleaving transactions badly)
- **D: Durability** (ensure that updates are not lost)

# Atomicity

- **Atomicity** means all steps execute or none of them
- **Why** would a transaction not fully execute?
  - System **crash** due to bugs or power failure
  - Transaction step violates integrity **constraints**
  - Explicit transaction **abort** (e.g., PG: ROLLBACK;)
- Internally, DBMS executes steps **separately**
  - May have to do **cleanup** in case of partial execution

# Consistency

- Consistency means data is consistent with all **constraints**
  - **Primary key** constraints
  - Referential integrity (**foreign key** constraints)
  - More **complex constraints** can be defined
- DBMS will **abort** transactions threatening consistency
- **Careful**, differs from distributed systems consistency

# Isolation

- Different users may execute transactions **concurrently**
  - E.g., bank has **many clients** transferring money
- Executing transactions sequentially is **inefficient**
  - E.g., can **overlap** idle times with other transaction
- DBMS may **interleave** steps from multiple transactions
- Isolation means **simulating** sequential execution to users

# Durability

- DBMS notifies users that transaction has **committed**
- Durability guarantees that committed updates **persist**
- This must hold even if DBMS or server **crashes**
- Must store enough info on disk to **restore** at startup
- Only notify user of commit **after** that has happened

# Outlook

- Atomicity **Next**
- Consistency **(Not discussed)**
- Isolation **Now!**
- Durability **Next**