Milestone 1

Dulce Torres, Kyle Pickle, Pranav Kode, Sean Nguyen, Ruqayyah Siddique
Our Project

**Data Model** stores the Page Ranges with their base page, tail page, the schema columns in **columnar form**.

**Bufferpool** maintains **data** in memory, has a page directory that **maps RIDs to pages in memory**.

**Query Interface** improves **discoverability** of data, through querying capabilities.
Page Range Flow

- Page
- Base page
- Tail page

Key:

- Page Range
- List of base pages
- List of tail pages

Base Page

- Schema
- RID
- Indirection

Tail Page

- Schema
- RID
- Indirection

Current Base Page

Current Tail Page
Page Directory routes RIDs to records

- **PageTable** stores all PageRanges in a multi-level table.
- **TLB** keeps a cache of recently-accessed records for quick access.
Bufferpool (TLB)

Translation Lookaside Buffer

- Simplified (Records already stored in memory)
- Speeds up access time
- Flexible # of rows and cols
- LRU replacement

RID | Record | Access Time
--- | --- | ---
... | ... | ...
... | ... | ...
... | ... | ...

Hash function (%)
- Multi-Level Page Table
- For M1, stores references to the actual records in memory
- Fixed-sized Pages make indexing much faster

- Multiple levels reduce storage cost with large #s of records
Index

- RHash
  - Hash Table + ordered linked list
  - Map column values to Nodes
  - Node
    - RID set
    - Next value

Hashing (55) = 9
Query API

- Work through functions defined in the table class
- **Delete**
  - Use the primary key to get the RID, delete from the database, then update the index if necessary
- **Insert**
  - Check if the primary key already exists before inserting
- **Select**
  - Get all RIDs containing the desired value, then locate the record for each RID.
- **Update**
  - Check if the primary key already exists
- **Sum**
  - Do a range based search based on the index keys, get the appropriate column from each RID, then sum the list
Performance on different hardwares