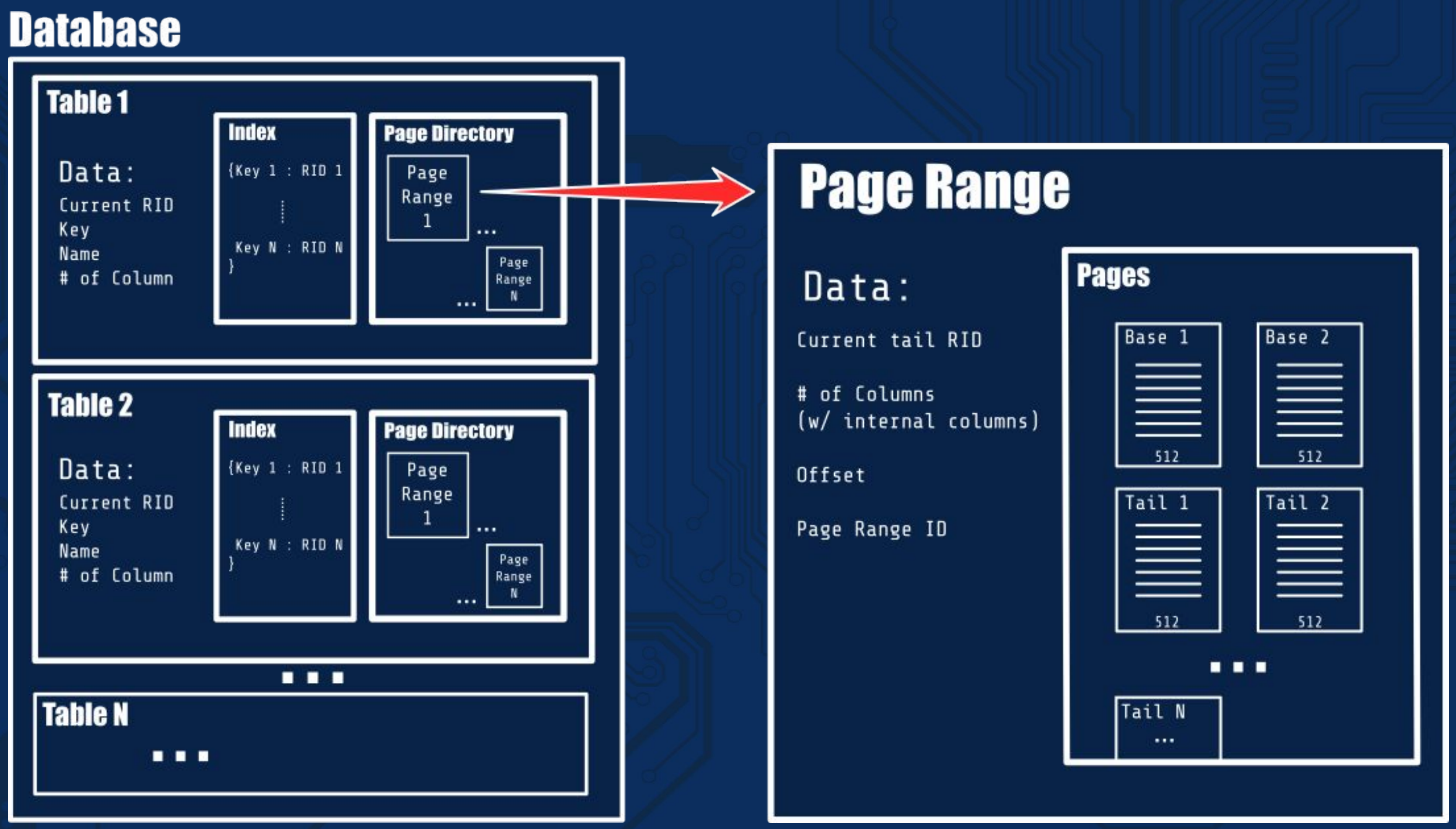
ECS165A

Milestone 1

Group 14

Ayman Dewan, Bin Lama, Nathan Ng, Kyle Insaurralde, Raynier Tan

Data Model



Database

Table 1

Data:

Current RID

Key

Name

# of Column

Index

{Key 1 : RID 1

⋮

Key N : RID N

}

Page Directory

Page Range 1 …

… Page Range N

Table 2

Data:

Current RID

Key

Name

# of Column

Index

{Key 1 : RID 1

⋮

Key N : RID N

}

Page Directory

Page Range 1 …

… Page Range N

■ ■ ■

Table N

■ ■ ■

Page Range

Data:

Current tail RID

# of Columns
(w/ internal columns)

Offset

Page Range ID

Pages

Base 1 512

Base 2 512

Tail 1 512

Tail 2 512

■ ■ ■

Tail N …

Bufferpool

Finding Page Range

```
page_range_index = int(rid // (BASE_PAGE_PER_RANGE
* NUM_PAGE_ENTRIES))
```

- We divide the RID by 1024 to find which page range a record is on. Since we set page range to have a max of 2 base pages
- i.e. a record with RID 2050, will be on page range 2 (3rd range)

Finding Base Page within Page Range

```
page_index = int(rid // NUM_PAGE_ENTRIES)
page_offset = int(rid % NUM_PAGE_ENTRIES)
```

- We divide the RID by 512 (amount of entries per page) to find which base page a record is on. Then find the offset (row) in which the record is
- i.e. RID 1036 will index will be on base page 2 (3rd total page, created in page range 1) and in row 12

Finding Tail Page Range

```
tail_rid = tail_rid | (self.page_range_id << TAIL_RID_BYTES)
page_range_index = int(rid >> TAIL_RID_BYTES)
```

- We find the page range of a tail RID by shifting it right by 7 bytes, since our whole tail RID follows this format: [PageRangeIndex][7 Byte Long Tail RID]

Finding Tail Page within Page Range

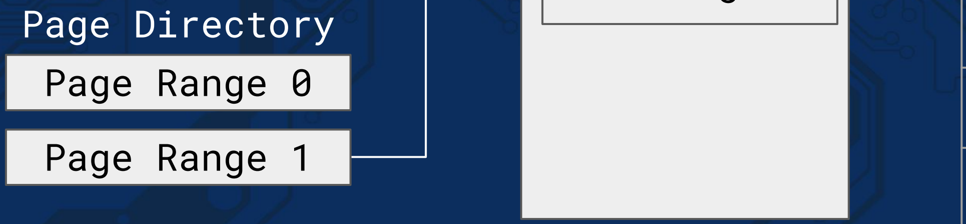
1. `rid &= 0xFFFFFFFFFFFFFF`
2. `page_index = int(rid // NUM_PAGE_ENTRIES)`
3. `page_offset = NUM_PAGE_ENTRIES - (rid % NUM_PAGE_ENTRIES) - 1`

- We find the page that the record is on, by first removing the largest byte (used for page ranges)
- Then, follow the same process as finding the base page. And then find the offset (which row on that particular tail page).

Query: Insert

1. Get table's current base RID, which would act as this new record's RID
2. Find the page range and base page the RID should belong to
3. Create a base record on the base page with offset determined by modulo
    a. Indirection column set to it's own RID
    b. Schema encoding initialized to 0
4. Add key to index

Insert([987,1]) with
cur RID = 1024



Page Directory

Page Range 0

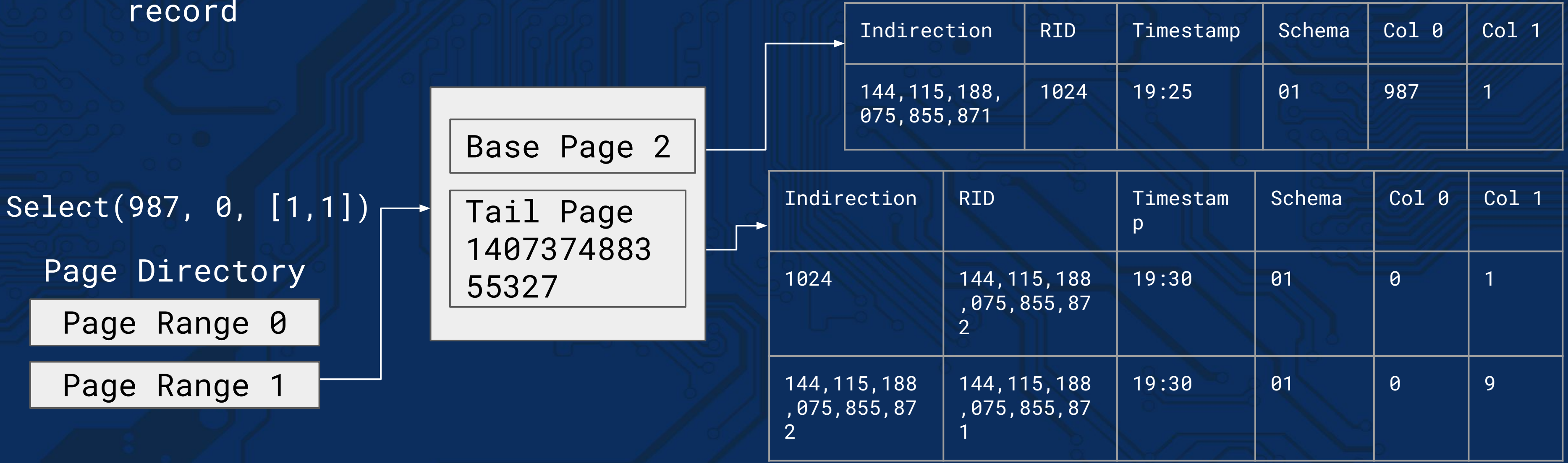
Page Range 1

Base Page 2

Indirection	RID	Timestamp	Schema	Col 0	Col 1
1024	1024	19:25	00	987	1

Query: Select

1. Check if key exist on index, else return False
2. Get the base record indirection and schema encoding
3. For each column queried, check schema to know if column has been updated
4. If column has been updated, read from tail record. Else, read from base record



Select(987, 0, [1,1])

Page Directory

Page Range 0

Page Range 1

Base Page 2

Tail Page 140737488355327

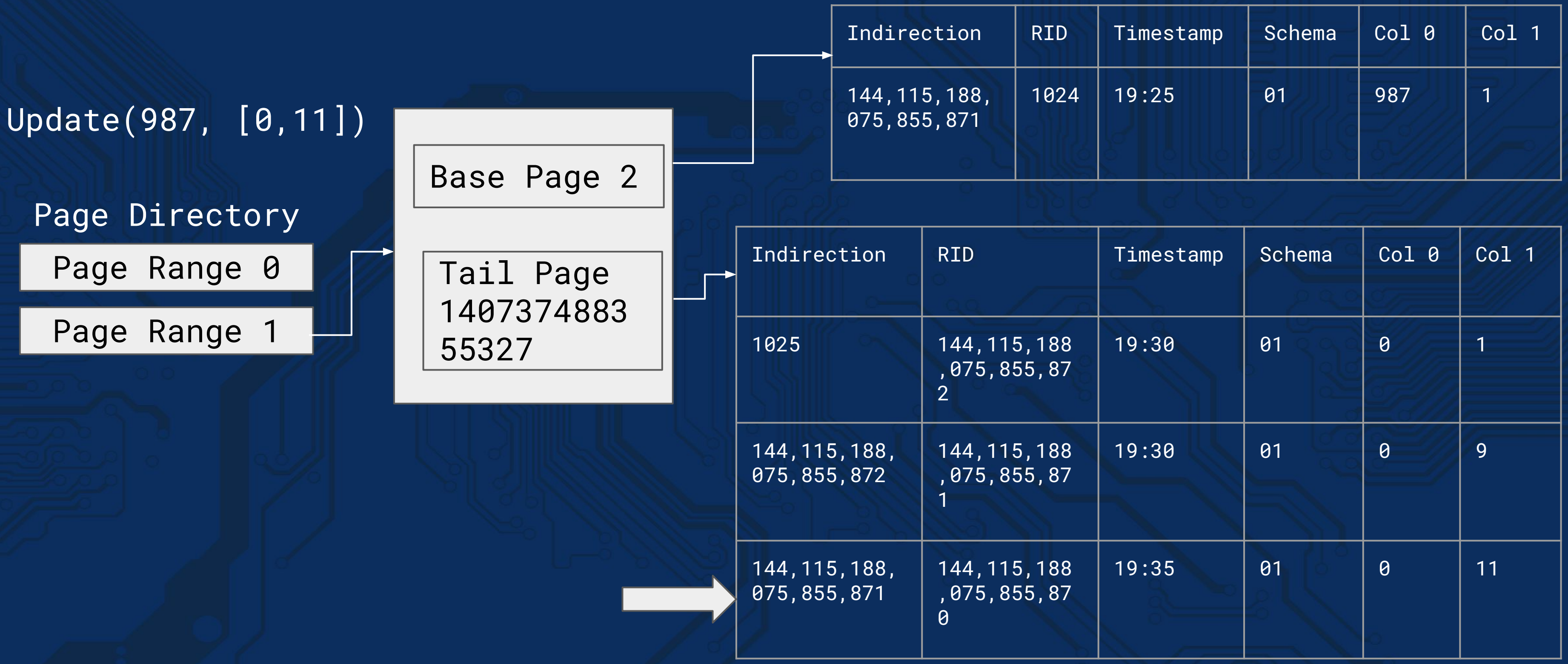
Indirection	RID	Timestamp	Schema	Col 0	Col 1
144,115,188,075,855,871	1024	19:25	01	987	1

Indirection	RID	Timestamp	Schema	Col 0	Col 1
1024	144,115,188,075,855,872	19:30	01	0	1
144,115,188,075,855,872	144,115,188,075,855,871	19:30	01	0	9

Query: Update

1. Check if key found in index, else return False
2. Get the key’s base record indirection and schema encoding
3. For each column to update, check schema if it has been updated before
4. If column not updated before, create a tail record with original column value
5. Create a tail record with:
    - Indirection pointing to latest tail record
    - Schema and column values reflecting all changes made including current update (Cumulative)
    - Columns that were not updated will be written as 0 (null)
6. Update base record’s indirection and schema

Query: Update (2)



Update(987, [0,11])

Page Directory

Page Range 0

Page Range 1

Base Page 2

Tail Page 140737488355327

Indirection	RID	Timestamp	Schema	Col 0	Col 1
144,115,188,075,855,871	1024	19:25	01	987	1

Indirection	RID	Timestamp	Schema	Col 0	Col 1
1025	144,115,188,075,855,872	19:30	01	0	1
144,115,188,075,855,872	144,115,188,075,855,871	19:30	01	0	9
144,115,188,075,855,871	144,115,188,075,855,870	19:35	01	0	11

Query: Sum

1. If no key found within range, return False
2. For each key found in range, use base RID to:
    a. Get schema to know which columns updated before
    b. Get indirection to find latest tail record
3. If schema for aggregate column is 1, append value retrieved from tail record
4. Else, append value retrieved from base record