Part 1 – Project Comprehension and Scope Analysis

CMPE321 Project 4: Dune Archive System

June 8, 2025

1 Operation Formats

1.1 Create Type

Format: `create type <type_name> <field_name>:<field_type> [<field_name>:<field_type>]* <primary_key>`
Sample Usage:

```
create type Movie title:string year:integer director:string title
```

Expected Result: Creates a new type definition with specified fields and primary key. The system stores this type definition for future record creation.

1.2 Create Record

Format: `create record <type_name> <field_name>=<value> [<field_name>=<value>]*`
Sample Usage:

```
create record Movie title="Dune" year=2021 director="Denis Villeneuve"
```

Expected Result: Creates a new record of the specified type with the given field values. The record is stored in the database and can be retrieved later.

1.3 Delete Record

Format: `delete record <type_name> <primary_key_value>`
Sample Usage:

```
delete record Movie "Dune"
```

Expected Result: Removes the record with the specified primary key value from the database.

1.4 Search Record

Format: `search record <type_name> <primary_key_value>`
Sample Usage:

```
search record Movie "Dune"
```

Expected Result: Retrieves and displays the record with the specified primary key value.

2 Primary Key Handling

2.1 Primary Key Specification

The primary key is specified as the last argument in the create type command. It must be one of the field names defined in the type.

2.2 Usage in Operations

- In search operations, the primary key value is used to uniquely identify the record to retrieve
- In delete operations, the primary key value is used to uniquely identify the record to remove

2.3 Constraints

- Uniqueness: Each record must have a unique primary key value within its type
- Data Type: The primary key field's type must be either string or integer
- Required: Every type must have exactly one primary key
- Immutable: Primary key values cannot be modified after record creation

3 Failure Case Summary

3.1 Type Definition Failures

- Duplicate type name
- Invalid field type specification
- Primary key not matching any field name
- Missing required fields

3.2 Record Operation Failures

- Type does not exist
- Duplicate primary key value
- Missing required fields
- Invalid field value type
- Record not found (for search/delete)

3.3 Failure Handling

- All failures are logged in log.csv
- No output is written to output.txt for failed operations
- The system continues processing subsequent operations

4 Log File Specification

The log.csv file maintains a persistent record of all operations with the following structure:

- Timestamp: ISO 8601 format (YYYY-MM-DD HH:MM:SS)
- Operation: The complete operation string
- Status: SUCCESS or FAILURE

The log file:

- Persists across multiple program runs
- Appends new entries to existing log
- Maintains chronological order of operations

5 Output Expectations

The output.txt file:

- Contains only successful search results
- Each record is displayed in a structured format
- Field values are shown in the order specified in type definition
- Failed searches are not written to output.txt
- File is overwritten at the start of each program run

6 Edge Case Scenarios

6.1 Common Edge Cases

- **Double Deletion:** Attempting to delete a record that was already deleted
  - System behavior: Logs failure, continues execution
  - Example: `delete record Movie "Dune"` (after first deletion)
- **Case-Sensitive Field Names:** Creating a type with field names that differ only in case
  - System behavior: Treats as duplicate fields, logs failure
  - Example: `create type Person name:string Name:string age:integer name`
- **Whitespace Handling:** Searching with trailing spaces in primary key
  - System behavior: Trims whitespace before comparison
  - Example: `search record Movie "Dune "`

7 Visual Overview of the File–Page–Record Structure

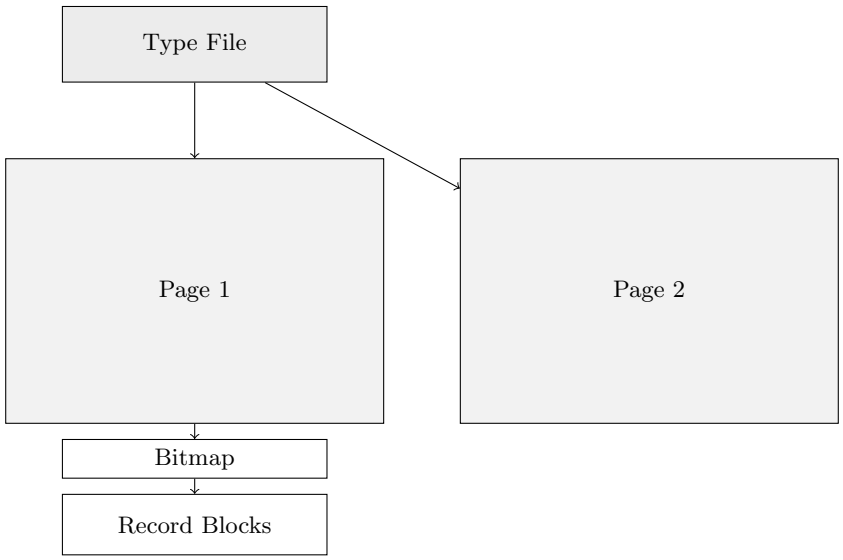

Figure 1: Expanded File–Page–Record Structure