Codebase Structure

- The codebase is organized into two main directories: **frontend** and **backend**.
- All data generated for databases, tables, and tuples is stored in the **database** directory, which is automatically created when the program is executed.
- The **frontend** directory contains all code related to command-line interface (CLI) inputs and outputs.
- The **backend** directory contains the core implementation of the storage manager.

Overall Layout of the data

All persistent data used and created by the system is stored inside the code/database/ directory. This directory serves as the root location for both metadata and table data. The folder structure and path constants defined in code/src/backend/layout specify how databases and tables are organized as directories and files within this location.

Database Directory Layout

```
database/
  ├── global/
  │    └── catalog.json
  └── base/
      ├── db1/
      │    └── {table}.dat
      ├── db2/
      │    └── {table}.dat
```

Directory Descriptions

- **database/**
  Root directory for all persistent data used and created by the system.

- **global/**
  Contains system-wide metadata required to interpret database structure.

- **catalog.json**
  Stores metadata for all databases, tables, and their column definitions.

- **base/**
  Contains one subdirectory per database.

- **{database}/**
  Represents a single database and holds all table files belonging to it.

- **{table}.dat**
  Physical file corresponding to a table, containing both table metadata and tuple data.

Catalog File Structure

The catalog is stored as a JSON file at database/global/catalog.json.

Catalog JSON Structure

```
{
  "databases": {
    "<database_name>": {
      "tables": {
        "<table_name>": {
          "columns": [
            {
              "name": "<column_name>",
              "data_type": "<data_type>"
            }
          ]
        }
      }
    }
  }
}
```

Example Catalog.json file

```
{
  "databases": {
    "users": {
      "tables": {
        "students": {
          "columns": [
            {
              "name": "id",
              "data_type": "INT"
            },
            {
              "name": "name",
              "data_type": "TEXT"
            }
          ]
        },
        "teachers": {
          "columns": [
            {
              "name": "id",
              "data_type": "INT"
            },
            {
              "name": "name",
              "data_type": "TEXT"
            }
          ]
        }
      }
    }
  }
}
```

Table File Structure

Each {table}.dat file stores table data as a contiguous sequence of bytes and is divided into fixed-size pages. Each page is 8 KB in size.

The first page of the file is reserved as the **Table Header**. Within this page, only the first 4 bytes are used to store the total number of pages that contain tuple data. The remaining bytes in the header page are currently unused.

All subsequent pages are data pages used to store tuples.

Page Structure

The page structure is based on the PostgreSQL slotted-page layout, with only the minimum required metadata implemented.

For reference, the PostgreSQL page layout is described at:
https://www.postgresql.org/docs/current/storage-page-layout.html#STORAGE-PAGE-LAYOUT-FIGURE

Each data page is divided into:

- A **page header**, which stores the lower and upper offsets
- An **Item ID array**, growing forward from the page header
- **Tuple data**, appended from the end of the page backward

The page-related implementation is located in src/backend/page/mod.rs.