PicoMite DB Engine: Command Reference

Platform: Raspberry Pi Pico 2 (RP2350)

1. Database Setup & Management These commands handle the creation and loading of database environments.

- **make-db <dbname> <csv_list>**
  - **Description:** Converts one or more CSV files into the engine's binary format (.dat) and creates the schema definition (.def). Automatically injects a "Soft Delete" flag (`delFlag`).
  - **Example:** `make-db us500 employees.csv, jobs.csv`
- **use-db <dbname>**
  - **Description:** Loads a database environment (tables and schema).
  - **Example:** `use-db us500`
- **ruler <filename>**
  - **Description:** Displays the first record of a file with a visual ruler. Useful for calculating fixed-width field positions manually.
  - **Example:** `ruler employees.csv`
- **EXIT**
  - **Description:** Closes all files and terminates the engine.

2. Data Retrieval (Native Mode) Syntax: `[filter] [ORDER BY field [DESC]] [SHOW fields | FORMAT name] [LIMIT n]` Queries are constructed by chaining clauses. If no command is recognized, the engine treats the input as a Filter.

- **Clauses**
  - **Filter Condition:** Standard comparison operators (=, >, <, >=, <=, <>). *Strings with spaces must be quoted.*
  - **Like Operator:** Use ~ for wildcard searches (`%` = multi-char, _ = single-char).
  - **SHOW <fields>:** Selects specific columns to display (comma-separated).
  - **FORMAT <name>:** Applies a saved view (see Tools section).
  - **ORDER BY <field> [DESC]:** Sorts the results. Default is Ascending.
  - **LIMIT <n>:** Restricts the number of results returned.
  - **TXT <file> / CSV <file>:** Exports the results to a text or CSV file on the SD card.
- **Examples**
  - **Simple Search:** `zip="90210"`
  - **Specific Columns:** `state="TX" SHOW city, phone`
  - **Sorting:** `age > 60 ORDER BY salary DESC`
  - **Wildcard:** `title ~ "Eng%"` (Matches Engineer, English, etc.)

- **Export:** `state="NY" CSV ny_export.csv`
- **Complex:** `age > 50 & state="CA" SHOW last, phone LIMIT 10`

3. Data Modification (CRUD) These commands physically alter the database files on the disk.

- **INSERT INTO <table> <fields...>**
  - **Description:** Appends a new record to the database.
  - **Example:** `INSERT INTO us500 first_name=John, last_name=Doe, city=ModelTown`
- **UPDATE <table> <ID> SET <field>=<value>**
  - **Description: Surgical Update.** Modifies a specific field in a single record identified by its ID. Rebuilds index immediately (unless Indexing is OFF).
  - **Example:** `UPDATE us500 42 SET salary=65000`
- **UPDATE <table> SET <field>=<value> WHERE <filter>**
  - **Description: Bulk Update.** Modifies a field for ALL records matching the filter. Rebuilds index only *once* after all updates are complete.
  - **Example:** `UPDATE us500 SET city=ModelCity WHERE city=ModelTown`
- **DELETE FROM <table>**
  - **Description: Soft Delete.** Marks records matching the filter as deleted (sets the internal `delFlag` to `*`). Deleted records are ignored by Indexer and Search.
  - **Syntax A (SQL):** `DELETE FROM us500 WHERE id=42`
  - **Syntax B (Native):** `[filter] DELETE FROM us500`
- **UNDELETE FROM <table>**
  - **Description:** Restores previously deleted records (sets `delFlag` to space).
  - **Syntax:** `[filter] UNDELETE FROM [table]`
  - **Example:** `city=modeltown UNDELETE FROM us500`

4. Performance & Transactions Controls the B-Tree Indexing system for speed and batch efficiency.

- **index <table> ON <field>**
  - **Description:** Creates/Rebuilds an index for the specified field.
  - **Example:** `index us500 ON zip`
- **index <field> IN <table>**
  - **Description:** Natural language alternative syntax.
  - **Example:** `index zip IN us500`
- **SET INDEXING OFF**
  - **Description:** Disables automatic index updates. Use this before running many INSERTs or UPDATEs to vastly improve speed.

- **SET INDEXING ON**
    - **Description:** Re-enables automatic index updates.
- **REINDEX**
    - **Description:** Commits any "Dirty" indexes (rebuilds them). Use this after a batch operation if Indexing was OFF.

5. Aggregates & Statistics Calculates statistics across the dataset.

- **COUNT [field] [WHERE filter]**
    - **Example:** `COUNT * WHERE state="TX"`
- **SUM [field] [WHERE filter]**
    - **Example:** `SUM salary WHERE title="Engineer"`
- **AVG [field] [WHERE filter]**
    - **Example:** `AVG age`
- **MIN / MAX**
    - **Example:** `MAX salary WHERE state="NY"`

6. Tools & Developer Hooks

- **RUN**
    - **Description:** Executes the hard-coded User Hook subroutine (`Sub UserCode` in the main program). Useful for testing hard-coded batch logic.
- **format <name> <fields>**
    - **Description:** Saves a list of fields as a reusable view definition (.fmt).
    - **Example:** `format contact firstname, lastname, email, phone`
- **batch <filename>**
    - **Description:** Executes a list of database commands from a text file sequentially.
    - **Example:** `batch nightly_jobs.txt`
- **test**
    - **Description:** Runs the `Test_Callback` subroutine to verify that your code is successfully intercepting database rows via `DB_Callback$`.
- **MODE SQL / MODE NATIVE**
    - **Description:** Switches the parser between Native syntax and a limited SQL-compatible subset.