CS 6360.003 – Database Design (Spring 2017)

DavisBase Programming Project

By,

Kruthika Vishwanath (kxv150930)

1. INTRODUCTION

1.1 PURPOSE

This document describes the high-level design for the Database Design Project for creation of a rudimentary DBMS namely `DavisBASE` which is a hybrid of both MySQL and SQLite.

1.2 SCOPE OF DOCUMENT

High Level Design Assumptions and Implementations.

Developmental Libraries, Tools used

1.3 DEVELOPMENTAL TOOLS USED

- Python 2.7.13
- Jython 2.7
- Cython 2.7
- JAVA 1.8
- MacOS Sierra

1.4 DESIGN ASSUMPTIONS

It is assumed that the database catalog information is present in two tables namely `davisbase_tables.tbl` and `davisbase_columns.tbl` Tables which are not user-modifiable. Also, DavisBASE is a rudimentary database and hence operation on only a single table. Though simple syntax checker is in place, it is always expected that the queries entered are Syntactically and Semantically correct and adherent to the DATA-TYPE that it is defined against. Not following the same might lead to unexpected result.

Only the following queries are supported.

. CREATE SCHEMA SCHEMA-NAME

. CREATE TABLE TABLE-NAME (COL1 DT1, [COL2 DT2...])

. USE SCHEMA-NAME

. DELETE FROM TABLE-NAME [WHERE]

    - Without WHERE deletes all entries from TABLE

. DROP TABLE TABLE-NAME

. UPDATE TABLE-NAME SET COLUMN=VALUE [WHERE]

    - Without WHERE update all entries of the corresponding COLUMN within constraint limitation

. INSERT INTO TABLE-NAME VALUES (V1, [V2...])

. DROP TABLE TABLE-NAME

. SHOW TABLES;

. SHOW DATABASES

. DROP DATABASE SCHEMA-NAME

Multi-Table queries (Joins) are not supported in this version of DavisBASE.

DATATYPES Supported ➢ TEXT, INT, BIGINT, REAL, DOUBLE, NULL, TINYINT, DATETIME, DATE/

KEYWORDS Reserved (Cannot be used by USER other than the intent its specified for)

- NULL, null, NOT NULL, PRIMARY KEY B*-Tree is used for saving and retrieving the data. An XML Configuration file is placed which is self explanatory. This file needs to be configured

first before compiling/linking or running the DavisBASE. Few user configurable parameters are placed uder dbase_Config/userConfig.xml file

`null`/`NULL` are reserved keywords and cannot be used. This means a field value is NULL (NOT BLANK). BLANK and NULL/null are different meaning in DavisBase. In short, an empty string is not NULL and a null string is not an empty string. Both are different. A Key with IS_NULLABLE field in `davisbase_columns.tbl` is `N' means inserting or updating that field's value with `null`/`NULL` is not permitted by DavisBASE and is rejected with appropriate error message. Similary a PRIMARY_KEY Field cannot be made nullable (in line with preserving ENTITY INTEGRITY CONSTRAINT).

TABLE_NAME and SCHEMA_NAME must be only valid Alphanumeric characters. Non-Alpha Numeric Characters are not allowed.

1.5 HIGH LEVEL DESIGN ARCHITECTURE

Apart from B*-Tree (B-Plus Tree) implemention, DavisBASE works in memory. This means all the B*-Tree pages are modified in memory and then are flushed onto disk.

For any Table that is created, a TABLE-NAME.tbl file will be created which uses an INTERNALLY GENERATED UNIQUE ROW-ID as its Key. It is to be noted that ROW-ID is auto-incremented by 1 by the software and is not user modifiable. User is not expected to CREATE a column named ROWID and or enter any value for ROWID when inserting data. Doing so might result in unpredictable result

Apart from a single `tbl` file, many separate idx files are created, one corresponding to each columns that the table contains. These idx files are secondary indices which complement the primary index which is a part of `tbl` file.

`ROWID` is not a separate 4 Byte field, rather it's always the first field of any payload that goes into the `tbl` file. The Right child pointer/ Right Sibling pointer is a 4 Byte hashed value of the page it points to. If only a single page is present, it is hashed to itself.

1.6 PRE-REQUISITE DIRECTORIES:

The file dbase_Config/userConfig.xml has to be modified to contain the the correct install path. Apart from this a directory named LOGS has to be created for capturing logs, if enabled in XML. All scripts must be present under a folder named `SOLUTION` and at the same level of `SOLUTION` is `LOGS` directory. It is expected that all the pre-requisite programs/developmental tools are to be installed and configured properly.

1.7 FEATURES:

- Support and Manipulation of Multiple Databases,
- USE, DROP DATABASE, SHOW DATABASE Commands.
- Handling of NULL Constraints for all applicable Column Data Types.
- Handling of Datetime formatting as per the requirement.
- DDL, DML and VDL supporting full handling of NULL Constraints
- LEX Parser for simple syntax validation rather than string tokenizer as former is much more efficicent and robust.
- The DavisBASE itself is not Case Sensitive but all values are stored in Upper case irrespective of the case of the input
- Supporting for B*-Tree (B+ Tree) Index Tables alongside Table's Table
- Self described Database (Catalog Files contains information about itself)
- Viewing information from `Catalog` of DavisBASE.
- Auto-Incrementing ROWID which is internally populated, isolated from user manipulation
- No strict requirement for having the first field as INT and mandatorily as PRIMARY KEY. This DavisBASE supports different column data types to be the first column of any table and it need not only be INT and PRIMARY KEY.
- Except for `idx` files, ROWID is the Internally Auto-Incremented populated PRIMARY KEY for every .tbl file
- B*-Tree is also implemented for `Catalog` Files.