API

**class Database**

Represents a Database that holds Tables.

Database()

Constructor function that creates an empty Database.

~Database()

Deconstructs the Database.

void addTable(Table, string)

Parameters: Table and a table name

Adds a Table to the Database with given name.

void dropTable(string)

Parameters: Table name with no whitespace

Deletes the Table with the given name from the Database.

vector<string> listTables()

Returns a vector of all the names of the Tables in the Database.

vector<Table> getTables()

Returns a vector of all the Tables in the Database.

Table query(string)

Parameters: query command, written in the form given below

Spaces required in query string between tokens

Ex: ( x != 2 )

Returns a Table when given a command of the form:

SELECT <argument>

FROM <argument>

WHERE <argument>

The SELECT argument allows either a list of which attribute names to keep or the symbol “\*” to indicate to keep all attributes.

The FROM argument will be a single Table name.

The WHERE argument will consist of:

* + - Comparisons (=, !=, >, <, >=, <=) between attributes and values of the same type
    - AND, OR, and NOT
    - Parentheses

void deleteQuery(string)

Parameters: query command, see above for form

Deletes tuples from a Table when given a command that follows the same structure as the query command.

**class Table**

Represents a Table that holds Records

Table()

Constructs a Table without any rows or columns.

Table(vector<AttributeTypeTuple>)

Constructs a Table from given vector of attribute names and types.

~Table()

Deconstructs the Table.

void add(AttributeTypeTuple)

Parameters: aTT

Adds a column to the end of the Table with the given attribute name and type.

void deleteATT(string)

Parameters: name of the attribute

Deletes the attribute with the given name from the Table.

void insert(Record)

Parameters: Record

Inserts the given Record into the Table.

vector<AttributeTypeTuple> getAttributes()

Returns a vector of the attribute names and types.

int getSize()

Returns the number of Records in the Table.

void rename(string, string)

Parameter: Original table name, then new table name

Changes the name of an attribute. The first parameter is the old name and the second name is the new name.

int sum(string)

Parameters: attribute

Returns the sums of the values of the selected attribute in the Table.

int count(string)

Parameters: attribute

Returns the count of non-null entries of the selected attribute in the Table.

int min(string)

Parameters: attribute

Returns the minimum value of the selected attribute in the Table.

int max(string)

Parameters: attribute

Returns the maximum value of the selected attribute in the Table.

Table crossJoin(Table&)

Parameters: other Table to be joined with

Returns one Table created by combining two Tables.

Record operator(int)

Parameters: Position in the values of the record to iterate to

Iterates through Records in the Table.

**class AttributeTypeTuple**

Represents a tuple that holds a string and a Type.

AttributeTypeTuple(string, string)

Parameters: attribute, string representation of a type

Constructs an AttributeTypeTuple with the values passed in.

~AttributeTypeTuple()

Deconstructs the AttributeTypeTuple.

string getAttribute()

Returns the string value.

string getType()

Returns the type value.

**class Record**

Represents a Record that holds data in string format.

Record()

Creates a Record without any data.

~Record()

Deconstructs the Record.

Record(vector<string>)

Parameters: vector of values to be added to the new record

Creates a Record with the given data entered.

void addValue(string)

Parameters: value to be added

Adds a value to the Record, the value is represented as a string.

void removeValue(int)

Parameters: position in the record that value should be removed from

Removes a value from the Record chosen by the given int.

string getValue(int)

Parameters: position in the record that the wanted value is at

Returns a value chosen by the given int.

string operator(int)

Parameters: position in the record to iterate to

Iterates through the string values in the Record.

**class TableIterator**

A class used to iterate through the Records of a Table.

TalbeIterator(Table)

Parameters: table name

Constructs a TableIterator to iterate through the given Table.

void next()

Moves the iterator to the next Record in the Table.

Record get()

Returns the Record that the iterator is currently on.

**class Type**

A class for declaring Type in an AttributeTypeTuple.

**class runtime\_error**

Outputs an error message in the form of a string if an error occurs.