

Implementation of Database Management System

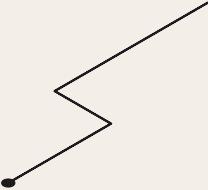
Option 3 UCB CS61B



Outline



- ① **Introduction**
- ② **Functionalities of our project**
- ③ **Code review**
- ④ **Real-time execution (the world cup)**
- ⑤ **Utilization from lectures and promotion**



Introduction



Introduction



● Purpose:

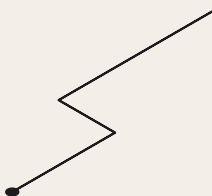
To implement a relational database management system (DBMS) with tables and a query language to manage the relational database and perform various operations on the data.

● Functions (For detail, please go to section 2 “functionalities”):

- Required by CS61B (creation, insertion, selection etc)
- Additional works (group by, having, order by etc)
- Return relevant messages if errors occur for the purpose of reminding

● Reflection:

Our understanding on the low-level working mechanism of database management system.



Functionalities



The basic implementation about DBMS

Required by CS61B

- Implement the Row class.
- Implement the table class.
- Implement the Database class.
- Implement insert and create.
- Implement select.

Additional works



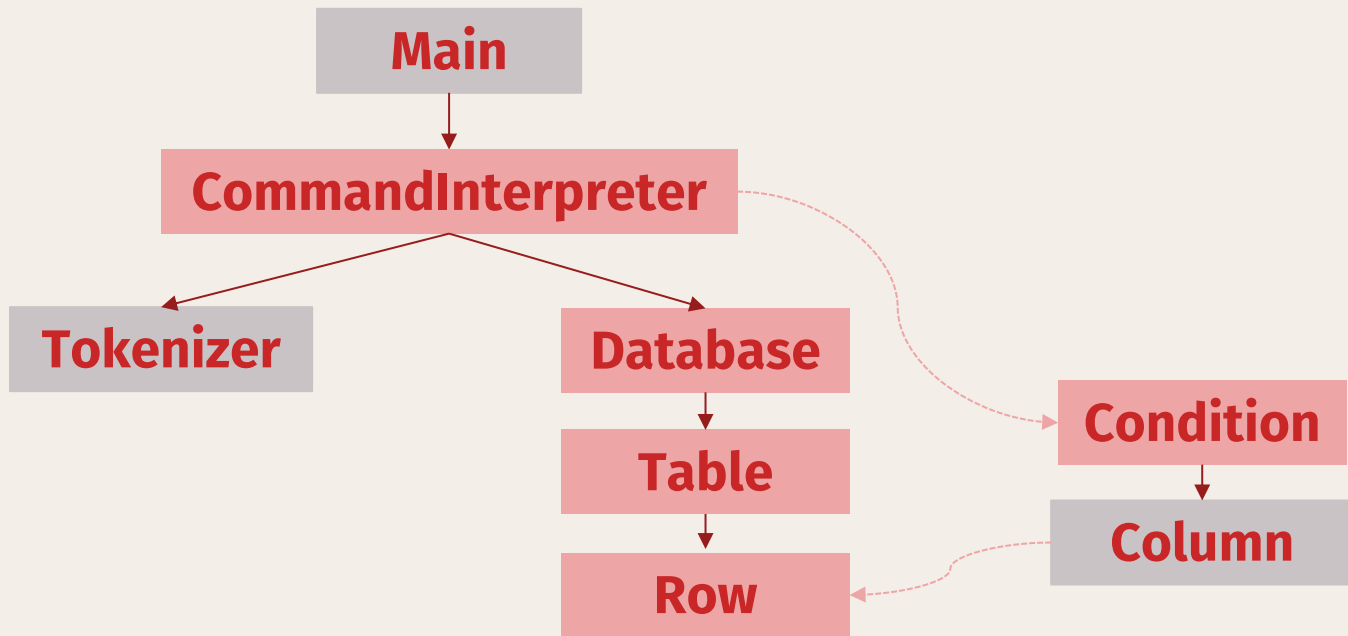
Add new features to the program

- Implement DELETE table operation.
- Implement ORDER BY.
- Implement GROUP BY and HAVING.
- Implement aggregate function (SUM, AVG, MAX, MIN, COUNT).
- Implement remove rows.
- Implement column minus and column plus. (our new ideas that MySQL does not support!)
- Improve visualization design of the printed table. (our new ideas that MySQL does not support!)

Code review



Project skeleton adapt the original backbone



Algorithm



1. Read the input
2. Check which type the statement is, and do corresponding operation

If the first string is “create” :

if the next string is “as”, let the select result as table return
else create an empty table with some columns

else if the first string is “load”: read the corresponding file and insert the read result as a table into the database

else if the first string is “exit” or “quit”, exit the program

else if the first string is “insert”, insert one row into a table

else if the first string is “delete”, delete one table from the database

else if the first string is “print”, print the table contents (visualization)

else if the first string is “select”:

1. read the column names (including aggregate functions)
2. read the table names (one or two) and deal with where condition if needed
3. deal with group by if needed
4. deal with aggregate functions if needed
5. deal with having condition if needed
6. deal with order by if needed
7. print the result (visualization)

else if the first string is “store”: store the contents of one table into one file

else if the first string is “column_plus”: plus two column into a new column (one table, data type is number)

else if the first string is “column_minus”: minus two column into a new column (one table, data type is number)

else if the first string is “remove_row”: remove some rows that satisfies the condition from one table

else return error message

3. Go to step 1

Real-time execution



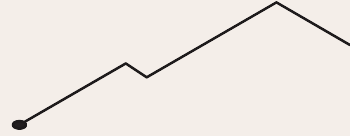


Let's use the data from the world cup to test out implementation!

Test **load** statement and **print** statement

```
load WorldCupGroups;  
load Shooters;  
print WorldCupGroups;
```

**I want to know the countries that
participate in the World Cup.**



Test **select** statement with **condition, group by, order by, aggregate functions**

```
select Country, Continent from WorldCupGroups;
```

```
select Continent, max(Goals_For) from WorldCupGroups group by Continent;
```

```
select Continent, avg(Goals_For) from WorldCupGroups group by Continent;
```

```
select Group, avg(Points), avg(Goals_For) from WorldCupGroups group by Group order by avg_Goals_For;
```

```
select Group, avg(Points), avg(Goals_For) from WorldCupGroups group by Group having avg_Goals_For > '2' order by avg_Goals_For;
```

I want to know the performance of different continents.



Test **select** statement with **condition** and **order by**

select Country, Continent, Wins, Goals_For, Goals_against from WorldCupGroups where Continent = 'Europe';
select Country, Continent, Goals_For from WorldCupGroups where Continent = 'Europe' order by Goals_For desc;

**I want to know the performance of
European countries**

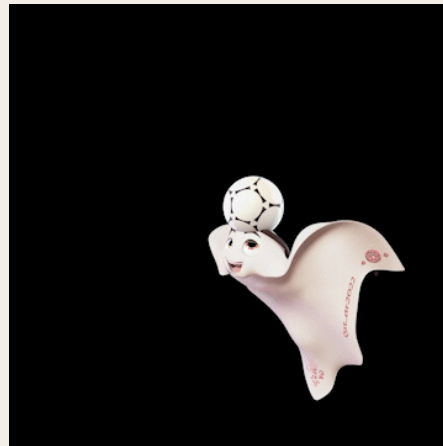


Test **column_minus** statement

column_minus WorldCupGroups: Goals_For and Goals_against to Goals_diff;



**I find that the table does not
contain goal difference, can we get
it?**

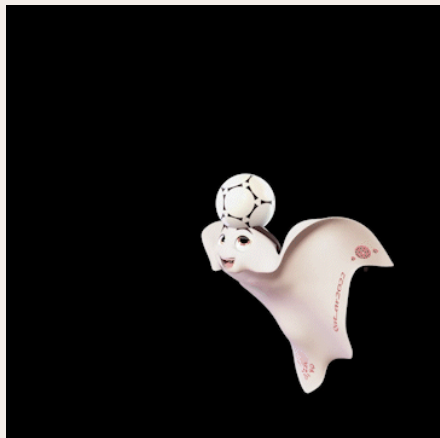


Test **select** statement from two tables

print Shooters;

select Country, Goals_For, Players, Player_Goals from WorldCupGroups, Shooters
where Country = Country;

**I know the shooters' list. Can you combine
the Top 10 players with their countries?**



Test **create** statement and **select** statement

create table ManyGoalsTeams as select Country, Goals_For from WorldCupGroups where Goals_For >= '4';

select Country, Goals_For from ManyGoalsTeams;

I love goals.

**Can you show me the countries
that have over 4 goals?**



Test **insert** statement and **remove_row** statement

insert into ManyGoalsTeams values 'China', '10';

select Country, Goals_For from ManyGoalsTeams;

remove_row from ManyGoalsTeams where Goals_For <= '9';

select Country, Goals_For from ManyGoalsTeams;

I love China!
Why no China??
(Imaginary Time)



Utilization and Promotion



Utilization from Lectures



- Data definition language (DDL): create
- Data query language (DQL): select -- where, group by, having, order by
- Data manipulation language (DML): insert, delete
- Aggregate functions: SUM, AVG, MAX, MIN, COUNT

Future Promotion



- Implement different data types for better performance of the DBMS
- Implement null value to differentiate null value and empty value
- Implement integrity constraints to make the database more accurate and reliable.

Thanks!



Group members :

Chi Gui

Jiayang Pang

Siyuan Zhao

Yuzheng Cong

Sipeng Sun

Jiale Zhong

