How about building a machine that analyze queries?

Constructor and functions:

```
$o=new MyDB("mybase");
```

mybase: table

functions

```
$o->add(field1,f2,f3,...);

@p=$o->delete("query1");
```

delete according to query1
query1: that's the query that first occurrence.

```
@p=$o->search("query2");
```

query2: that's the query that search first occurrence.

```
$o->print();
```

prints all database.

Then results:
Let's see how the queries can be defined
This query must return the first line matched. Does the DB file have to parsed once and store privately all results. Then pick ups in the list next result or parse again DB and then go to next result if DB not updated... We have as possibilities as we want but number is limited.

Synopsis:

query::=***SELECT*** <fields> ***FROM*** <tables> ***WHERE*** <tests>;

An idea:

```
$result=$o->select("fields that must be returned")->from("tables that must be parsed")->tests("tests on column of the DB");
```

Problems to solve:

- Checks if data base is not corrupted.

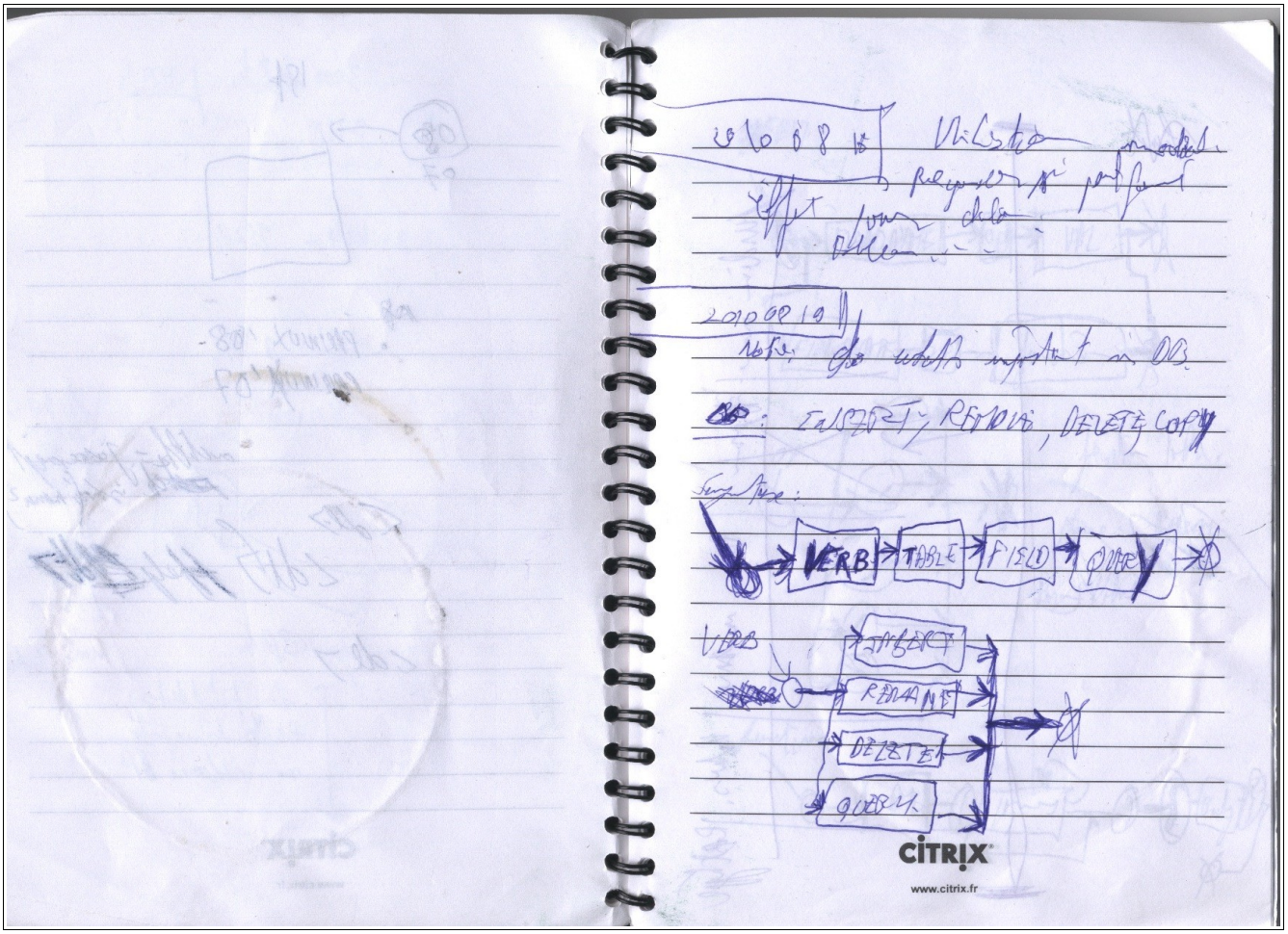
- Checks when processing update(writing in DB) manage a spool (first come first server (look file when created)). This can avoid the problem of the writer reader (is data when read still accurate?).

- Look problem of optimisation.

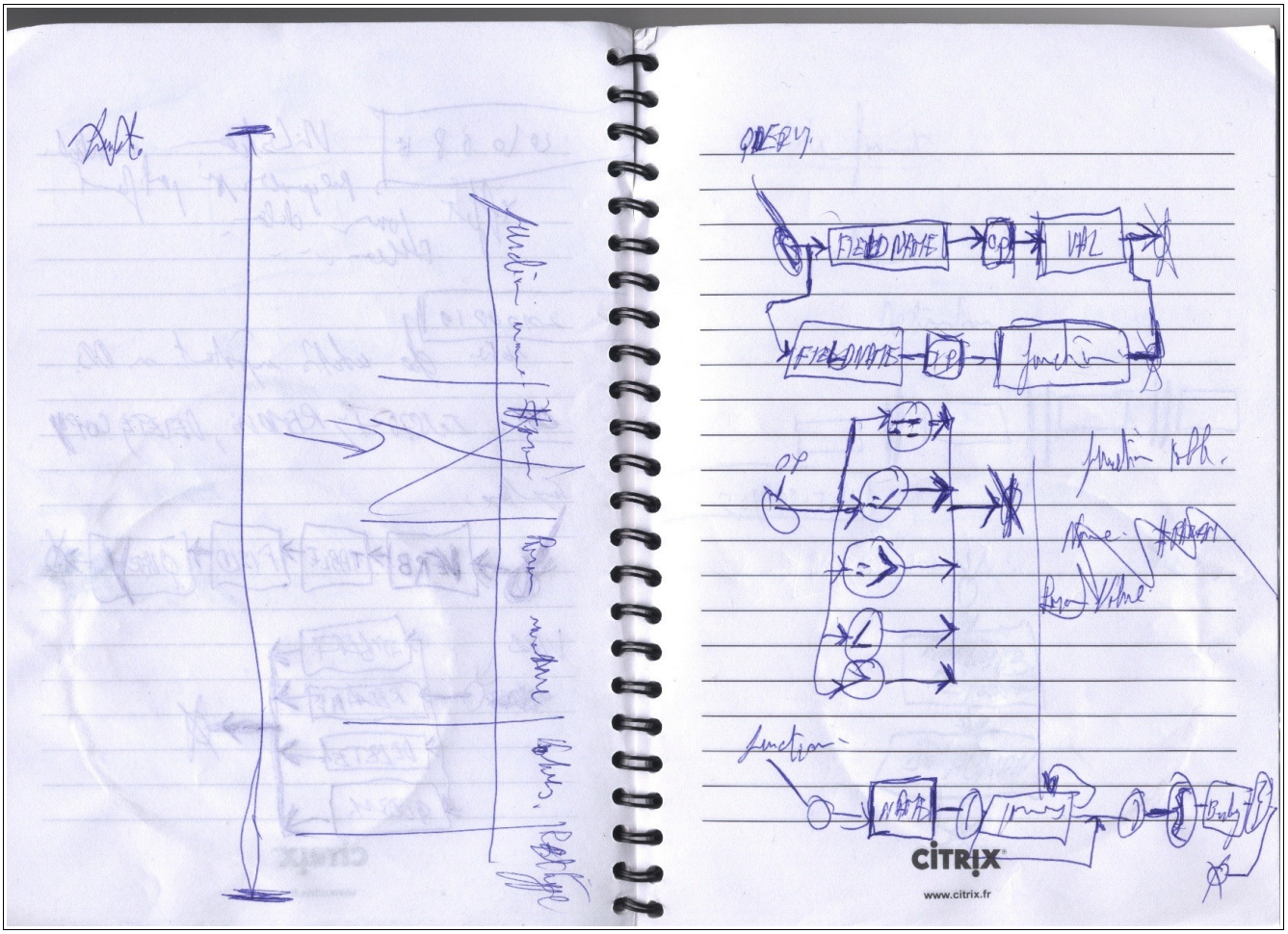
- The data base file format is not yet defined.

- There is not yet the problem of ***Marshalling*** involved. But it is a problem that have to be noted somewhere.

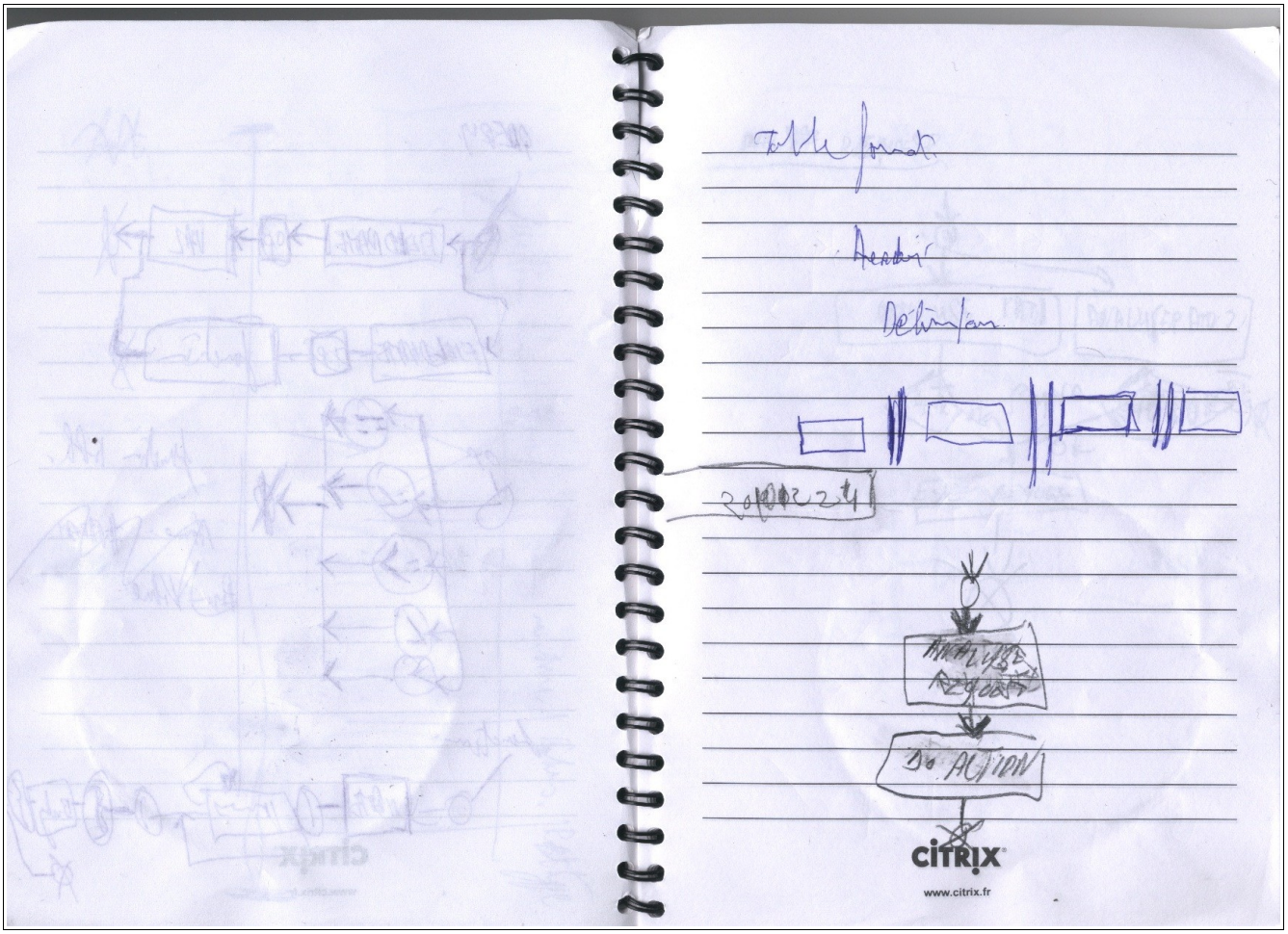
- Notes takes from the beginning:



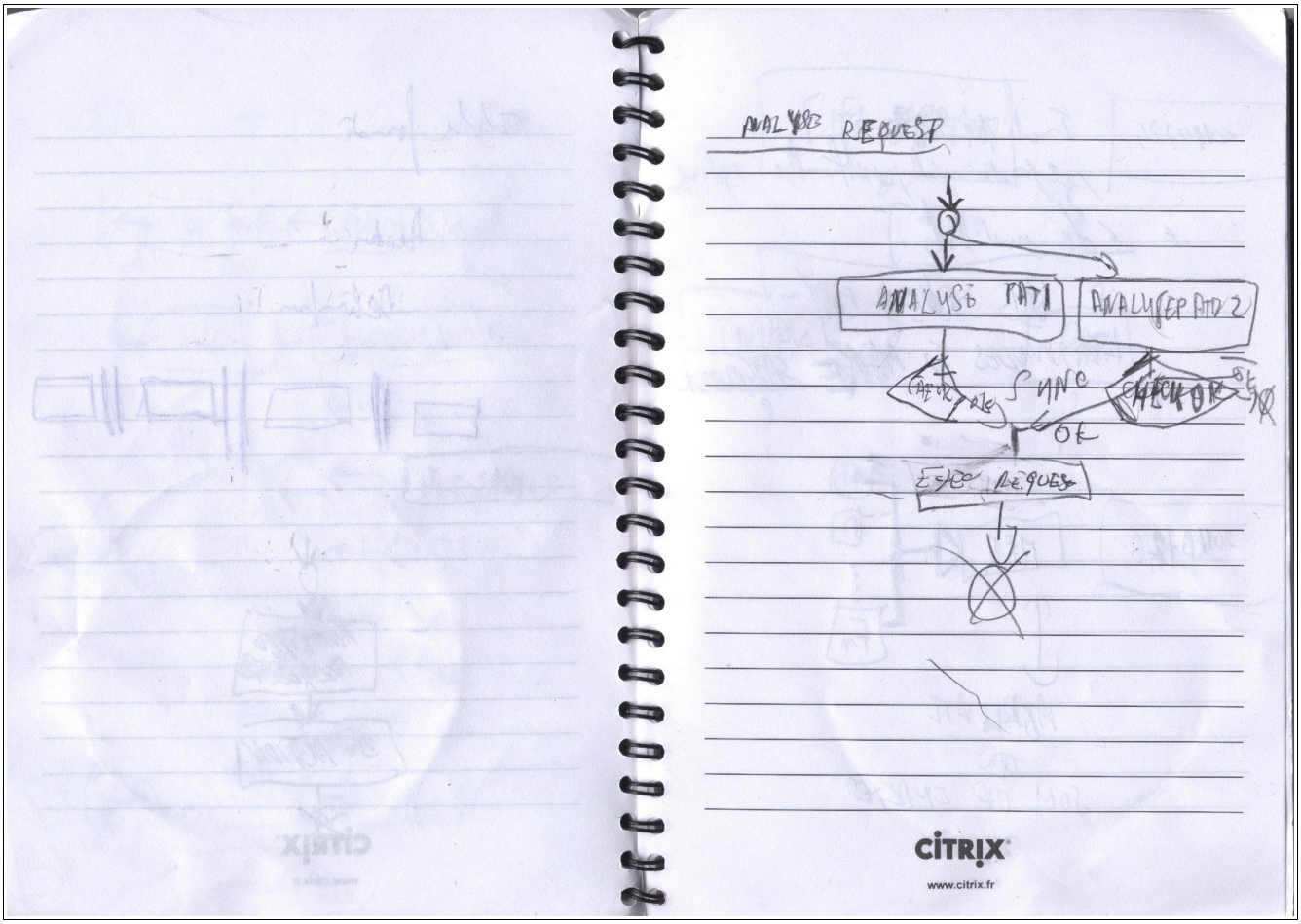
Primitive analyses?



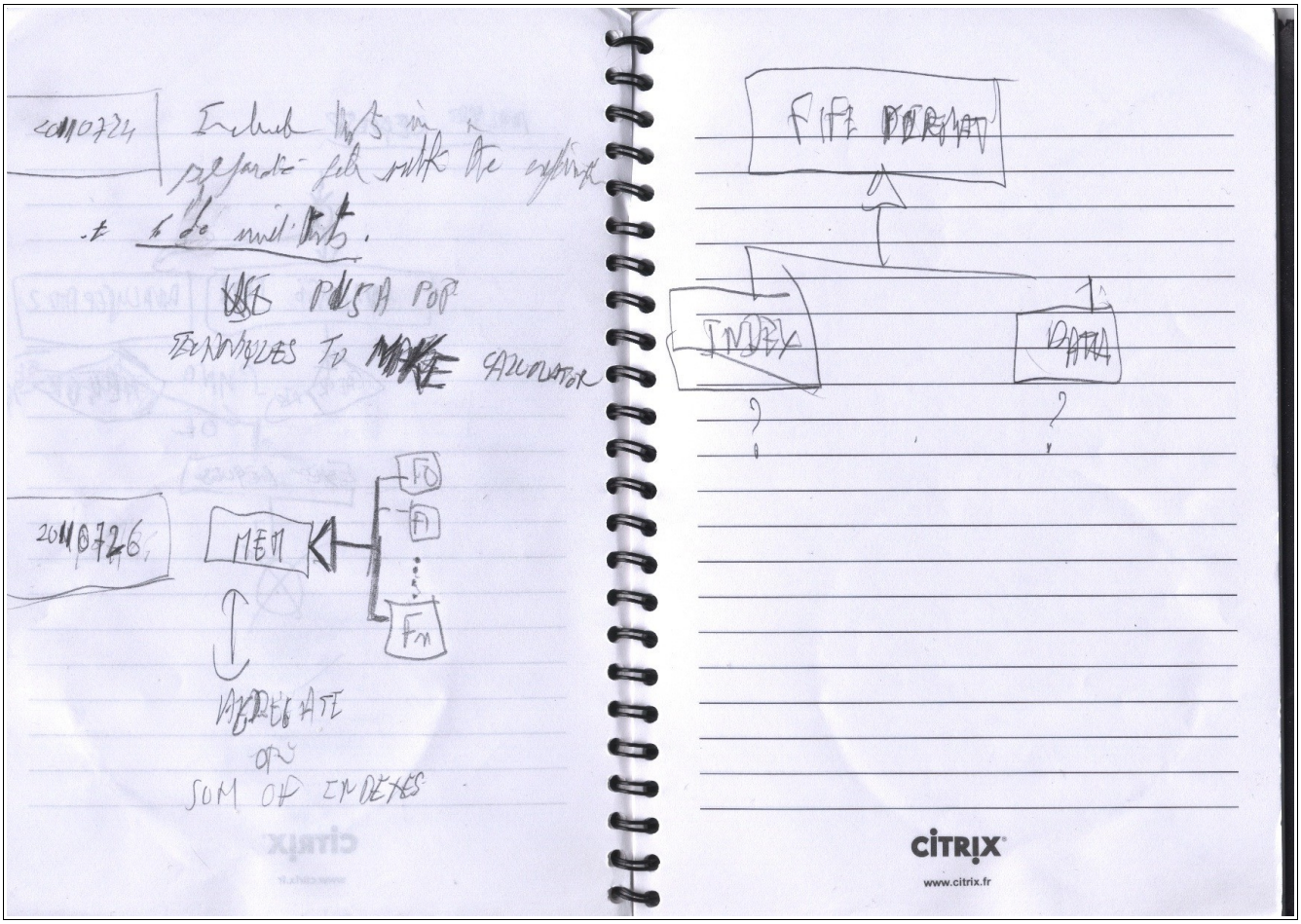
Query structure?



Query in action?



Any idea of the analyze query?



Any ideas on DB file structure?