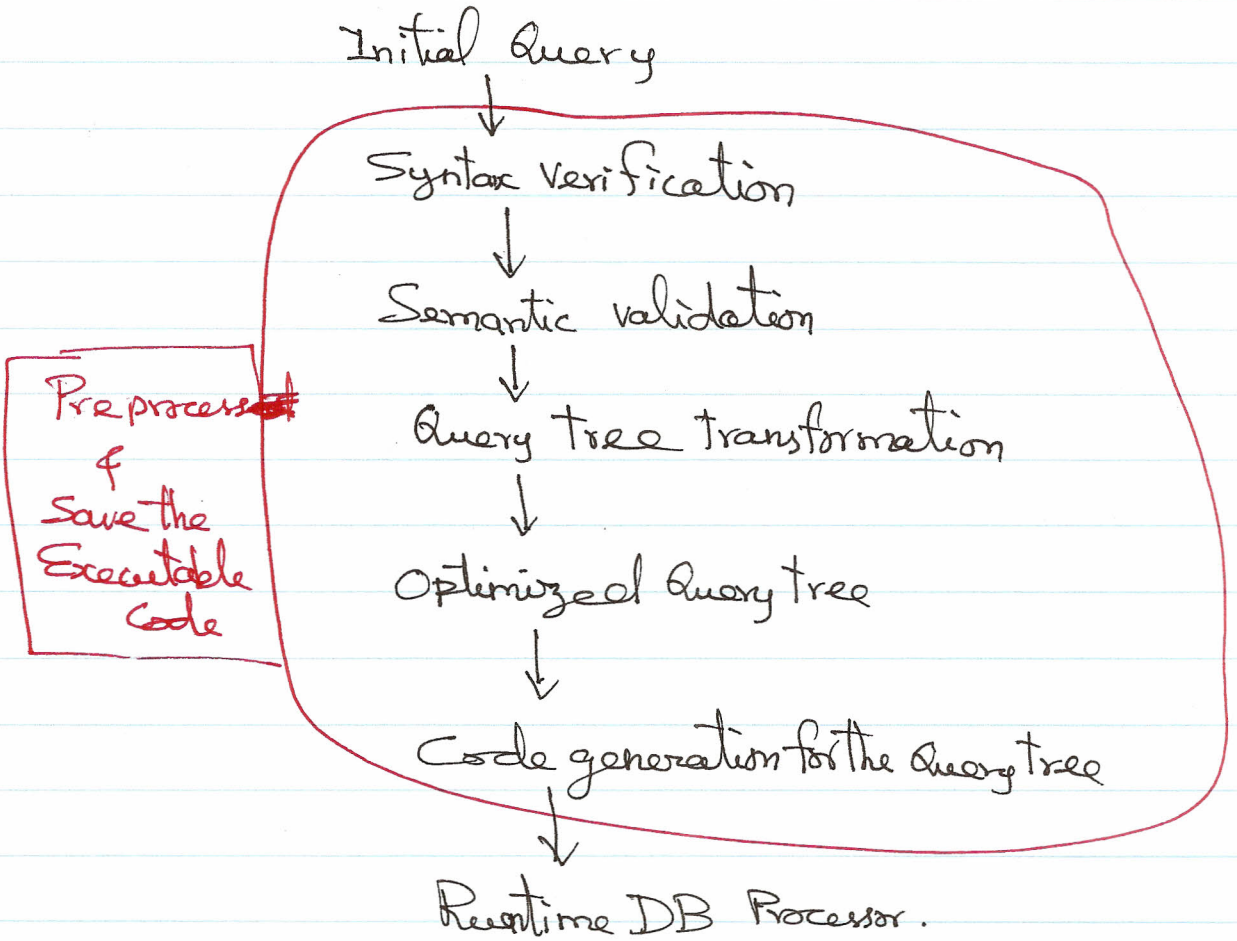
Query Execution Process



Stored Procedure

A procedure with several SQL statements and can accept parameters.
It also supports input/output operations within the procedure
Further local variables can be declared and used for passing values between SQL statements and data manipulations.
Additionally flow control such as if, while, can be supported.

* The procedure will be stored as an executable object within the DB.

Create procedure ~~uspEmpSalary~~
uspEmpTotalHours

parameter specifications

AS

Begin

Declaration of local variables

statements

End

- Stored procedures are preprocessed and saved as an executable object inside the DB.

- Later this executable SP object can be invoked.
  * interactively on the GUI interface
    Right click on the SP object
    select "Execute Stored Procedure..."
  * SQL Query Window
    EXEC SP_name parametervalue

=== CHECK constraint ===

In XAMPP, Check constraint needs to be specified next to the column in Table definition as
    sex char CHECK (sex IN ('F','M')),

=== Implementation of universal quantifiers ===

List the classes that has all students with gpa > 3.0
 = (there is no student in the class with (gpa <= 3.0)

However, the above condition will choose empty class (no student enrolled in) also in the list.

The correct semantics:
 = (the class has at least one student with gpa > 3.0)
   AND
   (there is no student in the class with (gpa <= 3.0)

=== Stored Procedure ===

For instructions, see
    C:\xampp\htdocs\course\scripts\company\storedproc\phpmyadmin\StoredProcNotes.txt
    C:\xampp\htdocs\course\scripts\company\storedproc\phpmyadmin\uspGetEmpSalary.sql

Execution through GUI
  Left panel: click on the company database, click Procedures
       click Execute (for the required stored procedure)
       Enter the Value field with input parameter string (no quotes)

Execution through SQL statements:
Left panel: click on the company database
click SQL tab

```
SET @inpArg1='333445555';
CALL uspEmpHours(@inpArg1);

SET @input='Wong';
SET @output='';
CALL uspGetEmpSalary(@input, @output);
SELECT @output AS Answer;
```

=== User management ===

```
CREATE USER 'new_user'@'localhost' IDENTIFIED BY 'password';
CREATE USER `mike`@`localhost` IDENTIFIED BY 'smith';
create user 'tim'@'localhost' identified by 'downey'
create user 'peter'@'localhost' identified by 'wong';

UPDATE mysql.user SET password = 'russell' WHERE user = 'tim';

DROP USER â€˜user_name`@â€˜localhost`;

SELECT User, Host FROM mysql.user; -- lists all users on the system
```

select * from mysql.user;
GUI: Click Home icon (UpperLeft), "User accounts" tab, "Edit privileges"

=== User privilege management ===

GRANT CREATE, SELECT ON * . * TO 'user_name'@'localhost';

List of Permission Types:
   CREATE â€” enable users to create a database or table
   SELECT â€” permit users to retrieve data
   INSERT â€” let users add new entries in tables
   UPDATE â€” allow users to modify existing entries in tables
   DELETE â€” enable users to erase table entries
   DROP â€” let users delete entire database tables

GRANT Privileges ON DatabaseName.TableName TO user@server;
FLUSH PRIVILEGES;

GRANT ALL PRIVILEGES ON *.* TO 'new_user'@'localhost';

grant select, update on company.works_on to 'tim'@'localhost';
grant select on company.* to 'mike'@'localhost';

REVOKE PERMISSION_TYPE ON database_name.table_name FROM â€˜user_nameâ€™@â€˜localhostâ€™;

SHOW GRANTS FOR 'user_name'@'localhost';

===