Pune Vidyarthi Griha's
College of Engineering, Nashik
206, Behind Reliance Petrol Pump, Dindoriroad, Mhasrul, Nashik-422004
Phone: 0253-6480000 / 36 /44 Toll Free Number: 18002665330
Website: https://pvgcoenashik.org Email: admission@pvgcoenashik.org
Approved by AICTE, New Delhi, DTE, Mumbai and
Affiliated to Savitribai Phule Pune University, Pune.
DTE Code: EN5330

" Structured Query Language"

By,

Prof. Anand N. Gharu

Asst. Professor

Computer Department,

PVG COE, Nashik.

Introduction

- Relation Database 1 : https://youtu.be/2KCObY8ixgw
- Relational Database : https://youtu.be/NqdZnYZ7Gvw

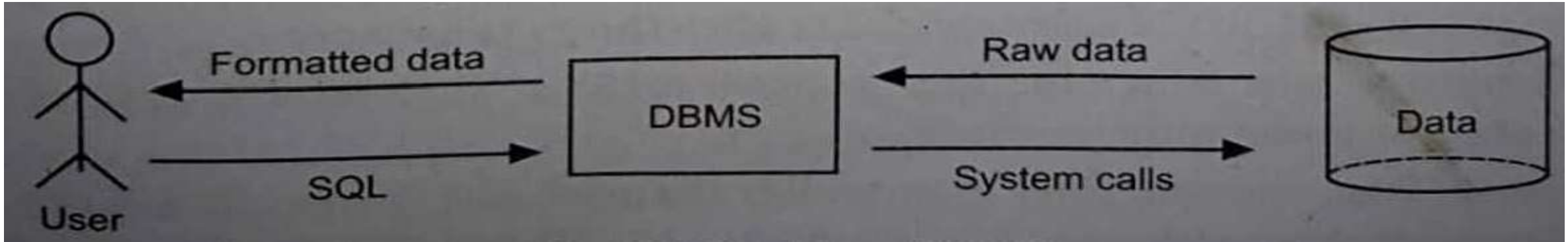
SQL

Live sql : https://www.w3schools.com/sql/trysql.asp?filename=trysql_op_in
Live sql : https://livesql.oracle.com/apex/f?p=590:1000

Introduction

Structured Query Language :

1. Structure Query Language(SQL) is a database query language used for storing, retrieving and managing data in Relational DBMS.
2. The Structured Query Language (SQL) is the set of instructions used to interact with a relational database.
3. SQL is a language to operate databases; it includes database creation, deletion, fetching rows, modifying rows, etc
4. Today almost all RDBMS(MySql, Oracle, Infomix, Sybase, MS Access) use SQL as the standard database query language.



Advantages of SQL

1. Faster Query Processing –

Large amount of data is retrieved quickly and efficiently. Operations like Insertion, deletion, manipulation of data is also done in almost no time.

2. No Coding Skills –

For data retrieval, large number of lines of code is not required. All basic keywords such as SELECT, INSERT INTO, UPDATE, etc are used and also the syntactical rules are not complex in SQL, which makes it a user-friendly language.

3. Standardised Language –

Due to documentation and long establishment over years, it provides a uniform platform worldwide to all its users.

Advantages of SQL

4. Portable –
It can be used in programs in PCs, server, laptops independent of any platform (Operating System, etc). Also, it can be embedded with other applications as per need/requirement/use.

5. Interactive Language –
Easy to learn and understand, answers to complex queries can be received in seconds.

6. Multiple data views –
Using the SQL language, the users can make different views of the database structure.

Disadvantages of SQL

1. Complex Interface –

SQL has a difficult interface that makes few users uncomfortable while dealing with the database.

2. Cost –

Some versions are costly and hence, programmers cannot access it.

3. Partial Control –

Due to hidden business rules, complete control is not given to the database.

Applications of SQL

- Allows users to access data in the relational database management systems.
- Allows users to describe the data.
- Allows users to define the data in a database and manipulate that data.
- Allows to embed within other languages using SQL modules, libraries & pre-compilers.
- Allows users to create and drop databases and tables.
- Allows users to create view, stored procedure, functions in a database.
- Allows users to set permissions on tables, procedures and views.

History of SQL

- 1970 − Dr. Edgar F. "Ted" Codd of IBM is known as the father of relational databases. He described a relational model for databases.
- 1974 − Structured Query Language appeared.
- 1978 − IBM worked to develop Codd's ideas and released a product named System/R.
- 1986 − IBM developed the first prototype of relational database and standardized by ANSI. The first relational database was released by Relational Software which later came to be known as Oracle

SQL Achitecture

- When you are executing an SQL command for any RDBMS, the system determines the best way to carry out your request and SQL engine figures out how to interpret the task.

- There are various components included in this process.

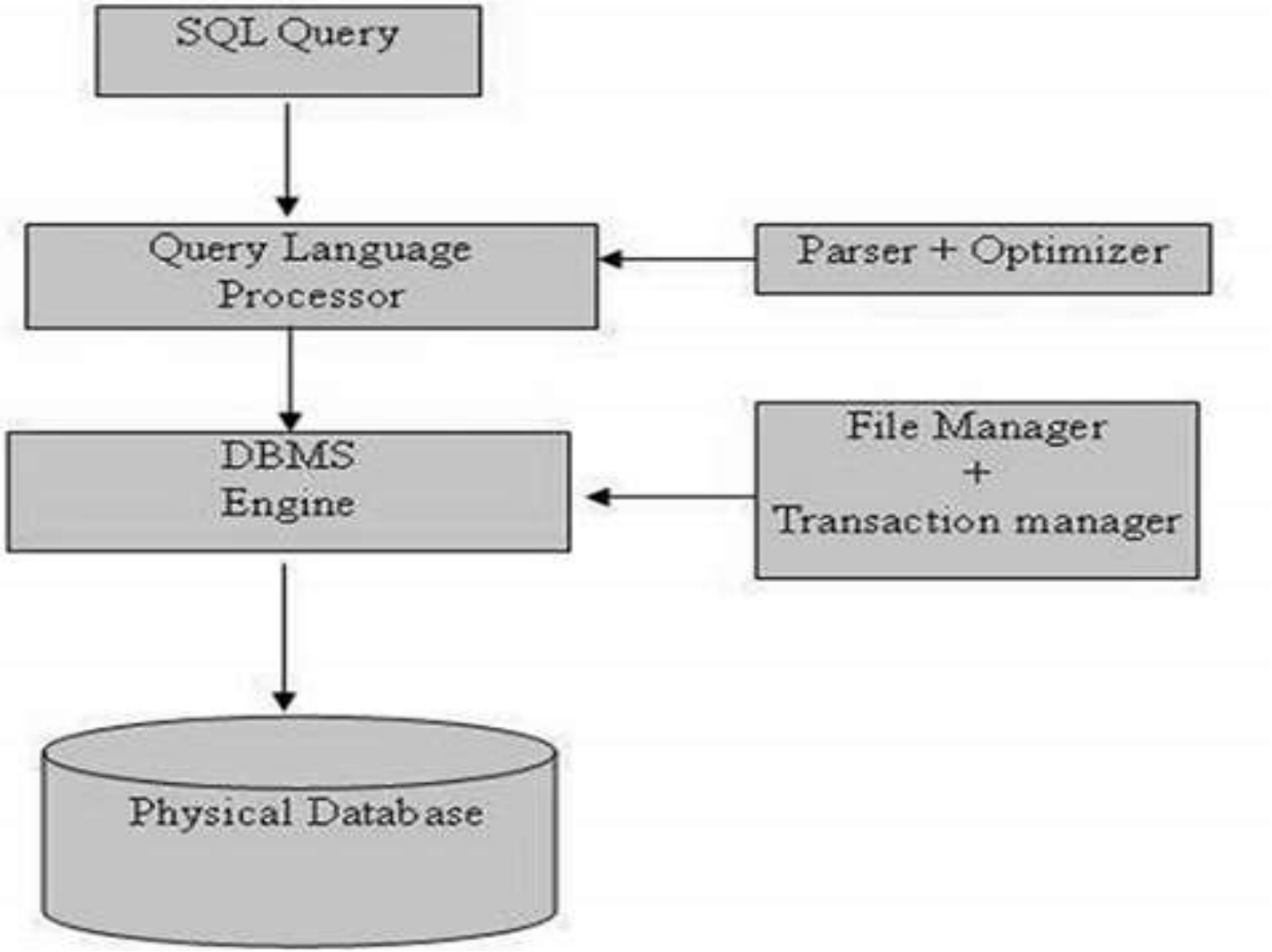
- These components are −

- Query Dispatcher
- Optimization Engines
- Classic Query Engine
- SQL Query Engine, etc.

A classic query engine handles all the non-SQL queries, but a SQL query engine won't handle logical files.

SQL Achitecture

A classic query engine handles all the non-SQL queries, but a SQL query engine won't handle logical files.



BASIC STRUCTURE OF SQL

1. Basic structure of an SQL expression consists of **select, from** and **where** clauses.
   - **select** clause lists attributes to be copied - corresponds to relational algebra **project**.
   - **from** clause corresponds to Cartesian product - lists relations to be used.
   - **where** clause corresponds to selection predicate in relational algebra.
2. Typical query has the form

$$\texttt{select}\ \ A_1, A_2, \ldots, A_n$$

$$\texttt{from}\ \ r_1, r_2, \ldots, r_m$$

$$\texttt{where}\ P$$

where each A_i represents an attribute, each r_i a relation, and P is a predicate.

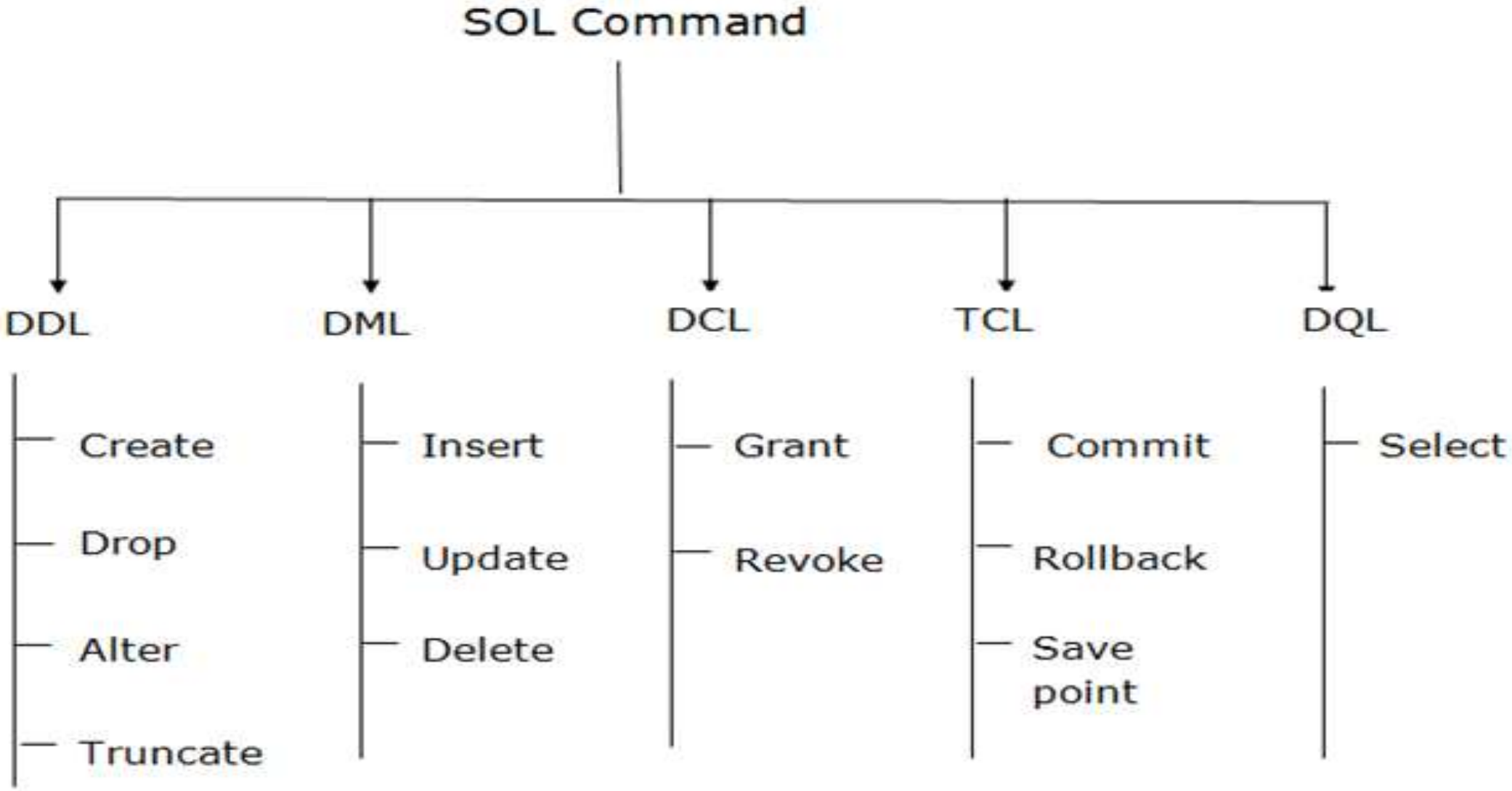
3. This is equivalent to the relational algebra expression

$$\Pi_{A_1, A_2, \ldots, A_n}(\sigma_P(r_1 \times r_2 \times \ldots \times r_m))$$

   - If the where clause is omitted, the predicate P is true.
   - The list of attributes can be replaced with a * to select all.
   - SQL forms the Cartesian product of the relations named, performs a selection using the predicate, then projects the result onto the attributes named.
   - The result of an SQL query is a relation.
   - SQL may internally convert into more efficient expressions.

Types of SQL Commands

There are five types of SQL commands: DDL, DML, DCL, TCL, and DQL.



Features/Component of SQL

SQL contains of some important features and they are:

1. Data Definition language (DDL):

- It contains of commands which defines the data.
- DDL changes the structure of the table like creating a table, deleting a table, altering a table, etc.
- All the command of DDL are auto-committed that means it permanently save all the changes in the database.

The commands are:

1. create: It is used to create a table.

Syntax:

```
create table tablename(attribute1 datatype.....attributen datatype);
```

2. drop: It is used to delete the table including all the attributes.

Syntax:

```
drop table tablename;
```

Features/Component of SQL

1. Data Definition language (DDL):

3. alter: alter is a reserve word which modifies the structure of the table.

Syntax:

alter table tablename add(new column1 datatype.....new columnx datatype);

4. rename: A table name can be changed using the reserver ‘rename’

Syntax:

rename old table name to new table name;

Features/Component of SQL

2. Data Manipulation Language (DML):

Data Manipulation Language contains commands used to manipulate the data.

The commands are:

- **insert:** This command is generally used after the create command to insert a set of values into the table.

Syntax:

insert into tablename values(attribute1 datatype);
:
:
:
insert into tablename values (attributen datatype);

Features/Component of SQL

- **delete:** A command used to delete particular tuples or rows or cardinality from the table.

Syntax:

delete from tablename where condition;

- **update:** It updates the tupples in a table.

Syntax:

update tablename set tupplename='attributename';

Features/Component of SQL

3. Triggers:

Triggers are actions performed when certain conditions are met on the data.A trigger contains of three parts.

(i). event – The change in the database that activates the trigger is event.

(ii). condition – A query or test that is run when the trigger is activated.

(iii). action – A procedure that is executed when trigger is activated and the condition met is true.

Features/Component of SQL

4. Client server execution and remote database access:
Client server technology maintains a many to one relationship of clients(many) and server(one). We have commands in SQL that control how a client application can access the database over a network.

5. Security and authentication:
SQL provides a mechanism to control the database meaning it makes sure that only the particular details of the database is to be shown the user and the original database is secured by DBMS.

6. Embedded SQL:
SQL provides the feature of embedding host languages such as C, COBOL, Java for query from their language at runtime

Features/Component of SQL

Transaction Control Language:
Transactions are an important element of DBMS and to control the transactions, TCL is used which has commands like commit, rollback and savepoint.

commit: It saves the database at any point whenever database is consistent.
Syntax:
commit;

rollback: It rollbacks/undo to the previous point of the transaction.
Syntax:
rollback;

savepoint: It goes back to the previous transaction without going back to the entire transaction.
Syntax:
savepoint;

Features/Component of SQL

Transaction Control Language:

Transactions are an important element of DBMS and to control the transactions, TCL is used which has commands like commit, rollback and savepoint.**commit:** It saves the database at any point whenever database is consistent.

Syntax:

commit;

rollback: It rollbacks/undo to the previous point of the transaction.

Syntax:

rollback; **savepoint:** It goes back to the previous transaction without going back to the entire transaction.

Syntax:

savepoint;

Advanced SQL:

The current features include OOP ones like recursive queries, decision supporting queries and also query supporting areas like data mining, spatial data and XML(Xtensible Markup Language).

SQL Commands

SQL Command

Live sql : https://www.w3schools.com/sql/trysql.asp?filename=trysql_op_in
Live sql : https://livesql.oracle.com/apex/f?p=590:1000

SQL defines following ways to manipulate data stored in an RDBMS.

DDL: Data Definition Language

- This includes changes to the structure of the table like creation of table, altering table, deleting a table etc.
- All DDL commands are auto-committed. That means it saves all the changes permanently in the database.

Command	Description
create	to create new table or database
alter	for alteration
truncate	delete data from table
drop	to drop a table
rename	to rename a table

SQL Commands

SQL Command

DML: Data Manipulation Language

- DML commands are used for manipulating the data stored in the table and not the table itself.

- DML commands are not auto-committed. It means changes are not permanent to database, they can be rolled back.

Command	Description
insert	to insert a new row
update	to update existing row
delete	to delete a row

SQL Commands

TCL: Transaction Control Language

These commands are to keep a check on other commands and their affect on the database. These commands can annul changes made by other commands by rolling the data back to its original state. It can also make any temporary change permanent.

Command	Description
commit	to permanently save
rollback	to undo change
savepoint	to save temporarily

SQL Commands

DQL: Data Query Language

Data query language is used to fetch data from tables based on conditions that we can easily apply.

Command	Description
select	retrieve records from one or more table

SQL Commands

DCL: Data Control Language

Data control language are the commands to grant and take back authority from any database user.

Command	Description
grant	grant permission of right
revoke	take back permission.

SQL Data Types

1. CHAR(Size)	It is used to specify a fixed length string that can contain numbers, letters, and special characters. Its size can be 0 to 255 characters. Default is 1.
2. VARCHAR (Size)	It is used to specify a variable length string that can contain numbers, letters, and special characters. Its size can be from 0 to 65535 characters.
3. INT(size)	It is used for the integer value.
4. INTEGER (size)	It is equal to INT(size).
5. FLOAT(p)	It is used to specify a floating point number.
6. DATE	It is used to specify date format YYYY-MM-DD. Its supported range is from '1000-01-01' to '9999-12-31'.
7. TIME(fsp)	It is used to specify the time format. Its format is hh:mm:ss. Its supported range is from '-838:59:59' to '838:59:59'

SQL Select Commnds

- The SELECT statement is used to select data from a database.
- The data returned is stored in a result table, called the result-set.

SELECT Syntax :

SELECT *column1*, *column2, ...column-n*

FROM *table_name*;

Here, column1, column2, ... are the field names of the table you want to select data from. If you want to select all the fields available in the table, use the following

Syntax: SELECT * FROM *table_name*; (e.g. select * from student;)

DDL COMMANDS

1. CRATE TABLE

1. **CREATE TABLE** creates a new table in the database.
2. The column parameters specify the names of the columns of the table.
3. The datatype parameter specifies the type of data the column can hold (e.g. varchar, integer, date, etc.).

Syntax :

```
CREATE TABLE table_name (
  column_1 datatype,
  column_2 datatype,
  column_3 datatype
);
```

e.g.
```
CREATE TABLE Persons (
    PersonID int,
    LastName varchar(20),
    FirstName varchar(20),
    Address varchar(20),
    City varchar(20)
);
```

2. DROP TABLE

The DROP TABLE statement is used to drop an existing table in a database.

Syntax
DROP TABLE table_name;

Example :

DROP TABLE student;

3. TRUNCATE TABLE

The TRUNCATE TABLE statement is used to delete the data inside a table, but not the table itself.

Syntax:
TRUNCATE TABLE table_name;

Example :

TRUNCATE TABLE Student;

4. ALTER TABLE

1. The ALTER TABLE statement is used to add, delete, or modify columns in an existing table.
2. The ALTER TABLE statement is also used to add and drop various constraints on an existing table.

Syntax:

ALTER TABLE *table_name* ADD *column_name datatype*;

ALTER TABLE table_name DROP COLUMN column_name;

ALTER TABLE table_name MODIFY COLUMN column_name datatype;

Example :

ALTER TABLE Customers ADD Email varchar(50);

5. RENAME TABLE

1. SQL RENAME TABLE syntax is used to change the name of a table.

Syntax:
RENAME old_table _name To new_table_name;

Example :

RENAME STUDENTS TO ANAND;

DML COMMANDS

1. INSERT TABLE

- It is possible to write the INSERT INTO statement in two ways.

The first way specifies both the column names and the values to be inserted:

Syntax:

```
INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);
```

```
INSERT INTO table_name
VALUES (value1, value2, value3, ...);
```

Example :

```
INSERT INTO Customers (CustName, Contact, Address, City, Country)
VALUES (‘Anand', ‘8087777708', ‘samarth nagar', ‘Nashik', ‘India');
```

```
INSERT INTO Customers (CustomerName, City, Country)
VALUES (‘Anand', ‘Nashik', ‘India');
```

2. UPDATE TABLE

- The UPDATE statement is used to modify the existing records in a table.

Syntax:

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

Example :

```
UPDATE Customers
SET Custname = 'Aanand', City= ‘Dhule'
WHERE CustomerID = 1;
```

3. DELETE TABLE

- The DELETE statement is used to delete existing records in a table.

Syntax:

DELETE FROM table_name WHERE condition;

Example :

DELETE FROM Customers WHERE CustomerName=‘Anand';

DQL COMMANDS

SQL Select Commnds

- The SELECT statement is used to select data from a database.
- The data returned is stored in a result table, called the result-set.

SELECT Syntax :

SELECT *column1*, *column2, ...column-n*

FROM *table_name*;

Here, column1, column2, ... are the field names of the table you want to select data from. If you want to select all the fields available in the table, use the following

Syntax: SELECT * FROM *table_name*; (e.g. select * from student;)

TCL COMMANDS

1. COMMIT COMMAND

- Transaction control language or TCL commands deal with the transaction within the database.

1. Commit

This command is used to save all the transactions to the database.

Syntax:

Commit;

Example :

```
DELETE FROM Students
WHERE RollNo =25;
COMMIT;
```

2. ROLLBACK COMMAND

2. Rollback :
Rollback command allows you to undo transactions that have not already been saved to the database.

Syntax:
Rollback;

Example :

DELETE FROM Students
WHERE RollNo =25;

3. SAVEPOINT COMMAND

3. Savepoint
Savepoint command allows user to save the session for temporary

Syntax:
SAVEPOINT SAVEPOINT_NAME;

Example :

SAVEPOINT RollNo;

DCL COMMANDS

1. GRANT COMMAND

- Transaction control language or TCL commands deal with the transaction within the database.

Examples of DCL commands:

1. Grant
2. Revoke

1. Grant :

This command is use to give user access privileges to a database.

Syntax:

GRANT SELECT, UPDATE ON MY_TABLE TO SOME_USER, ANOTHER_USER;

Example :

GRANT SELECT ON Users TO'Tom'@'localhost;

2. REVOKE COMMAND

2. Revoke :

- It is useful to back permissions from the user.

Syntax:

REVOKE privilege_nameON object_nameFROM {user_name |PUBLIC |role_name}

Example :

REVOKE SELECT, UPDATE ON student FROM BCA, MCA;

Group by & Order by COMMANDS

1. Group by COMMAND

The GROUP BY Statement in SQL is used to arrange identical data into groups with the help of some functions. i.e if a particular column has same values in different rows then it will arrange these rows in a group.

Important Points:

- GROUP BY clause is used with the SELECT statement.
- In the query, GROUP BY clause is placed after the WHERE clause.
- In the query, GROUP BY clause is placed before ORDER BY clause if used any.

1. Group by COMMAND

Syntax:

```
SELECT column1, function_name(column2)
FROM table_name
WHERE condition
GROUP BY column1, column2
ORDER BY column1, column2;
```

function_name: Name of the function used for example, SUM() , AVG().

table_name: Name of the table.

condition: Condition used.

1. Group by COMMAND

Example : group by Single column

```
SELECT NAME, SUM(SALARY) FROM Employee
GROUP BY NAME;
```

Example : group by Multiple column

```
SELECT SUBJECT, YEAR, Count(*)
FROM Student
GROUP BY SUBJECT, YEAR;
```

2. Order by COMMAND

The ORDER BY statement in sql is used to sort the fetched data in either ascending or descending according to one or more columns.

By default ORDER BY sorts the data in ascending order.

- We can use the keyword DESC to sort the data in descending order and the keyword ASC to sort in ascending order.
- **Syntax :**

SELECT * FROM table_name ORDER BY column_name ASC|DESC

table_name: name of the table.

column_name: name of the column according to which the data is needed to be arranged. **ASC**: to sort the data in ascending order.

DESC: to sort the data in descending order.

| : use either ASC or DESC to sort in ascending or descending order

2. Order by COMMAND

Sort according to single column: In this example we will fetch all data from the table **Student** and sort the result in descending order according to the column **ROLL_NO**.

e.g. SELECT * FROM Student ORDER BY ROLL_NO DESC;

Sort according to multiple columns:

In this example we will fetch all data from the table Student and then sort the result in ascending order first according to the column Age. and then in descending order according to the column ROLL_NO.

e. g. SELECT * FROM Student ORDER BY Age ASC , ROLL_NO DESC;

Aggregate Function in SQL

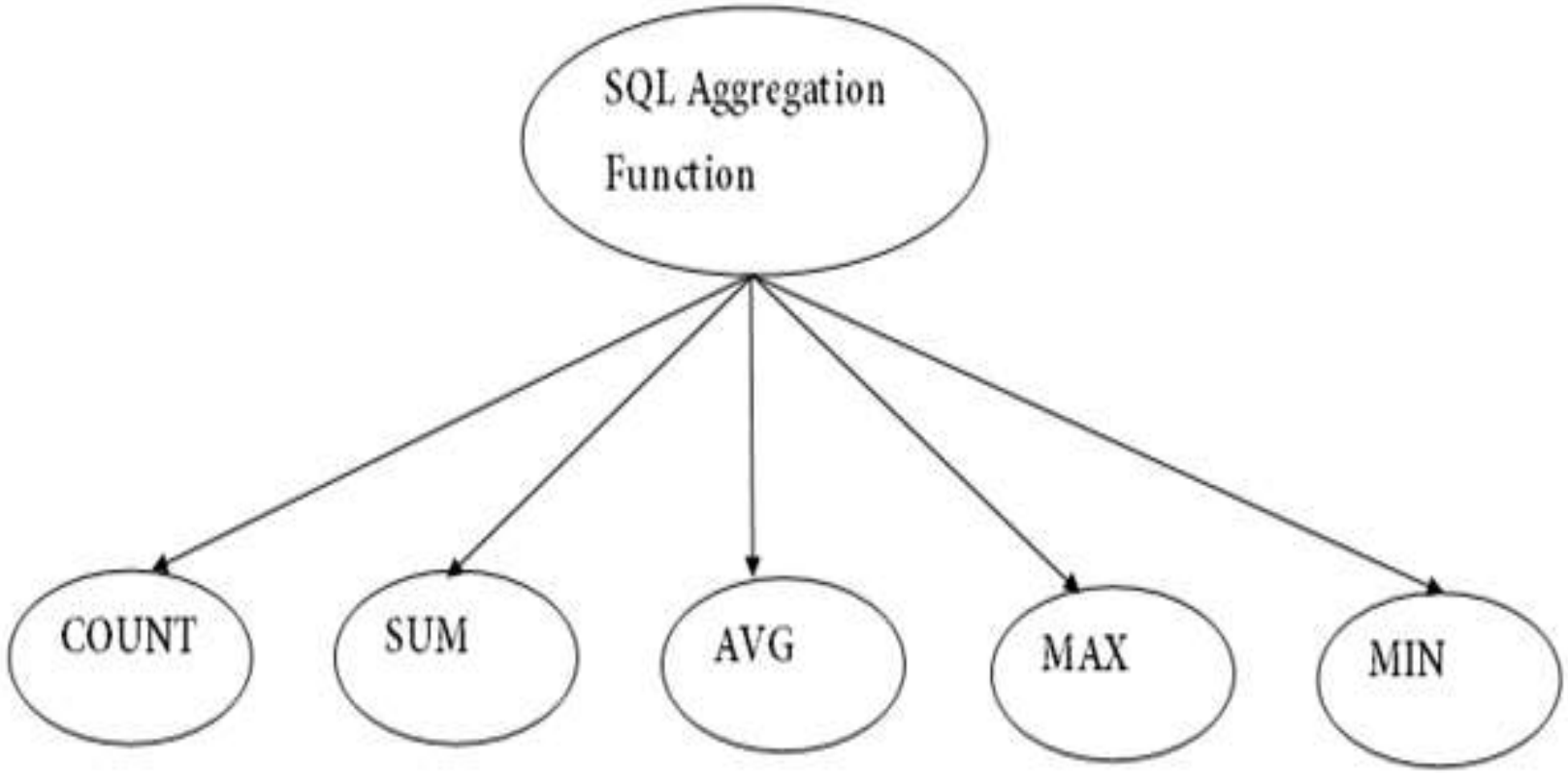
A. N. Gharu

Aggregate Function

In database management an aggregate function is a function where the values of multiple rows are grouped together as input on certain criteria to form a single value of more significant meaning.

Various Aggregate Functions

1) Count()
2) Sum()
3) Avg()
4) Min()
5) Max()



Aggregate Function

1. COUNT FUNCTION :

- COUNT function is used to Count the number of rows in a database table. It can work on both numeric and non-numeric data types.
- COUNT function uses the COUNT(*) that returns the count of all the rows in a specified table. COUNT(*) considers duplicate and Null.

Syntax

COUNT(*)
or
COUNT([ALL|DISTINCT] expression)

Example: COUNT()
1) SELECT COUNT(*) FROM PRODUCT;

2) COUNT(*) FROM PRODUCT;
    WHERE RATE>=20;

Aggregate Function

2. SUM Function :

Sum function is used to calculate the sum of all selected columns. It works on numeric fields only.

Syntax :

SUM()

or

SUM([ALL|DISTINCT] expression)

Example: SUM()

SELECT SUM(COST) FROM PRODUCT;

Aggregate Function

2. SUM Function :

Sum function is used to calculate the sum of all selected columns. It works on numeric fields only.

Syntax :

SUM()

or

SUM([ALL|DISTINCT] expression)

Example: SUM()

```
SELECT SUM(COST) FROM PRODUCT;

SELECT SUM(COST) FROM PRODUCT
WHERE QTY>3;
```

Aggregate Function

3. AVG function :

The AVG function is used to calculate the average value of the numeric type. AVG function returns the average of all non-Null values.

Syntax :

AVG()
or
AVG([ALL|DISTINCT] expression)

Example:

SELECT AVG(COST) FROM PRODUCT;

Aggregate Function

4. MAX Function :

MAX function is used to find the maximum value of a certain column. This function determines the largest value of all selected values of a column.

Syntax :

MAX()
or
MAX([ALL|DISTINCT] expression)

Example:

SELECT MAX(RATE) FROM PRODUCT; ;

Aggregate Function

5. MIN Function :

MIN function is used to find the minimum value of a certain column. This function determines the smallest value of all selected values of a column.

Syntax :

MIN()
or
MIN([ALL|DISTINCT] expression)

Example:

SELECT MIN(RATE) FROM PRODUCT;

String Operations in SQL

STRING OPERATION IN SQL

Database consists of some string values. The most commonly used operation on string is pattern matching. The operator used for pattern matching is **Like.**

The patterns are case sensitive, i.e. uppercase characters do not match lowercase characters or vice versa.

With the keyword **like**, we can use two characters:

Percent (%):

It matches with any substring. It means that % stands for more number of characters.

Underscore (_):

It matches with any single character. It means _ stands for one character only.

Consider following examples.

(a) "mad%": It matches with any string which begins with "**mad**". It may have characters after "mad".

For example: madagascar, madake, madam, madhura etc.

(b) "%mi%": It matches any string containing "**mi**" in it. The result may have characters before and after it.

For example: millar, milly, smith, etc.

(c) "---": matches with any string which has exactly 3 characters in it.

For example: jam, sam, mat etc.

(d) "_ a _": It matches with any string which has exactly 3 characters and the middle character is "a".

For example: cat, bat, mat etc.

Natural Join & Cartesian Join in SQL

NATURAL JOIN IN SQL

Natural Join is a type of Inner join which is based on column having same name and same datatype present in both the tables to be joined.

The syntax for Natural Join is,

SELECT * FROM

table-name1 NATURAL JOIN table-name2;

Example :

SELECT * from class NATURAL JOIN class_info;

NATURAL JOIN IN SQL

Example of Natural JOIN

Here is the **class** table,

ID	NAME
1	abhi
2	adam
3	alex
4	anu

and the **class_info** table,

ID	Address
1	DELHI
2	MUMBAI
3	CHENNAI

OUTPUT NATURAL JOIN

Natural join query will be,

```
SELECT * from class NATURAL JOIN class_info;
```

The resultset table will look like,

ID	NAME	Address
1	abhi	DELHI
2	adam	MUMBAI
3	alex	CHENNAI

CARTESIAN PRODUCT IN SQL

Cartesian Product in DBMS is an operation used to merge columns from two relations. Generally, a cartesian product is never a meaningful operation when it performs alone. However, it becomes meaningful when it is followed by other operations. It is also called Cross Product or Cross Join.

The syntax for Natural Join is,

Table1 X Table2

CARTESIAN PRODUCT

Syntax of Cartesian product (X)

```
R1 X R2
```

Cartesian product (X) Example

Table 1: R

```
Col_A     Col_B
-----     ------
AA         100
BB         200
CC         300
```

Table 2: S

```
Col_X     Col_Y
-----     -----
XX         99
YY         11
ZZ         101
```

Query:

Lets find the cartesian product of table R and S.

```
R X S
```

Output:

```
Col_A     Col_B     Col_X     Col_Y
-----     ------    ------    ------
AA         100      XX         99
AA         100      YY         11
AA         100      ZZ         101
BB         200      XX         99
BB         200      YY         11
BB         200      ZZ         101
CC         300      XX         99
CC         300      YY         11
CC         300      ZZ         101
```

OUTPUT NATURAL JOIN

Natural join query will be,

```
SELECT * from class NATURAL JOIN class_info;
```

The resultset table will look like,

ID	NAME	Address
1	abhi	DELHI
2	adam	MUMBAI
3	alex	CHENNAI

CREATE RDB IN 3NF

EXAMPLE - 1

Attempt the following :

Consider the following entities and their relationship :

Game (gno, gname, no-of-player, coachname)

Player (pno, pname)

Game and player are related with many-to-many relationship

Create RDB in 3NF and solve the following queries using SQL (any five) :

(a) Delete a row from Game table for game "Cricket"

→ Delete from Game

where gname = "Cricket"

Attempt the following :

Consider the following entities and their relationship :

Game (gno, gname, no-of-player, coachname)

Player (pno, pname)

Game and player are related with many-to-many relationship

Create RDB in 3NF and solve the following queries using SQL (any five) :

(b) Display all players who play game "Table Tennis"

→ Select pname, gname from Players p , Game g

where p.pno = g.gno and gname = "Table Tennis";

Attempt the following :

Consider the following entities and their relationship :

Game (gno, gname, no-of-player, coachname)

Player (pno, pname)

Game and player are related with many-to-many relationship

Create RDB in 3NF and solve the following queries using SQL (any five) :

(c) List all games played by Rajesh

→ SELECT gname
FROM Game , Player
WHERE pname = "Rajesh" and Game.gno = Player.pno;

Attempt the following :

Consider the following entities and their relationship :

Game (gno, gname, no-of-player, coachname)

Player (pno, pname)

Game and player are related with many-to-many relationship

Create RDB in 3NF and solve the following queries using SQL (any five) :

(d) Add column Age in the player table

→ Alter table Player add(Age int);

Attempt the following :

Consider the following entities and their relationship :

Game (gno, gname, no-of-player, coachname)

Player (pno, pname)

Game and player are related with many-to-many relationship

Create RDB in 3NF and solve the following queries using SQL (any five) :

(e) **Count total no. of players whose coach is "Kiran"**

→ Select count(no-of-player) from Game
where coachname = "Kiran"

Attempt the following :

Consider the following entities and their relationship :

Game (gno, gname, no-of-player, coachname)

Player (pno, pname)

Game and player are related with many-to-many relationship

Create RDB in 3NF and solve the following queries using SQL (any five) :

(f) Count max no. of players in a game.

→ Select max(no-of-player) from Game;

EXAMPLE - 2

9.**Attempt the following**

Consider the following entities and their relationship :

Item (item no, name, quantity)

Sup (no, name, addr, city, phone-no)

Item and sup are related with many-to-many relationship with rate, discount.

Constraints : primary key and item qty > 5 and rate > 0

Create RDB in 3NF and write queries in sql (any five) :

(a) Insert a row in item table.

→ Insert into item values(001 , “abc”, 10);

9.**Attempt the following**

Consider the following entities and their relationship :

Item (item no, name, quantity)

Sup (no, name, addr, city, phone-no)

Item and sup are related with many-to-many relationship with rate, discount.

Constraints : primary key and item qty > 5 and rate > 0

Create RDB in 3NF and write queries in sql (any five) :

(b) Find the rate and discount of the item mouse.

→ Select rate , discount from Item i, Sup s

where i.item no= s.no and name = “mouse”,

9.**Attempt the following**

Consider the following entities and their relationship :

Item (item no, name, quantity)

Sup (no, name, addr, city, phone-no)

Item and sup are related with many-to-many relationship with rate, discount.

Constraints : primary key and item qty > 5 and rate > 0

Create RDB in 3NF and write queries in sql (any five) :

(c) Count the number of items supplied by supplier “Mr. Navathe”.

→ Select count(quantity) from Item i, sup s

where i.item no = s.no and name = “Mr. Navathe”;

9.**Attempt the following**

Consider the following entities and their relationship :

Item (item no, name, quantity)

Sup (no, name, addr, city, phone-no)

Item and sup are related with many-to-many relationship with rate, discount.

Constraints : primary key and item qty > 5 and rate > 0

Create RDB in 3NF and write queries in sql (any five) :

(d) Display the details of all suppliers from 'Pune' city.

→ Select * from Sup

where city = "Pune";

9.**Attempt the following**

Consider the following entities and their relationship :

Item (item no, name, quantity)

Sup (no, name, addr, city, phone-no)

Item and sup are related with many-to-many relationship with rate, discount.

Constraints : primary key and item qty > 5 and rate > 0

Create RDB in 3NF and write queries in sql (any five) :

(e) Display item name in ascending order.

→ Select * from Item

Order by name ASC;

9.**Attempt the following**

Consider the following entities and their relationship :

Item (item no, name, quantity)

Sup (no, name, addr, city, phone-no)

Item and sup are related with many-to-many relationship with rate, discount.

Constraints : primary key and item qty > 5 and rate > 0

Create RDB in 3NF and write queries in sql (any five) :

(f) Display supplier name in descending order of quantity.

→ Select name from Sup s, Item i

Where i.item no = s.no

order by quantity DESC;

EXAMPLE - 3

8.Attempt the following :

Consider the following entities and relationships.

Item (I_no, I_name, I_qty)

Po (P_no, P_date)

Supplier (S_no, S_name, S_addr)

Item and Po are related with one to many relationships along with descriptive cost and quantity.

Supplier and Po are related with one-to-many relationships.

Create a RDB for the above and solve the following queries :

(i) Insert a row in Item table.

→ Insert into Item values(001 , ‘xyz’ , 10);

8.Attempt the following :

Consider the following entities and relationships.

Item (I_no, I_name, I_qty)

Po (P_no, P_date)

Supplier (S_no, S_name, S_addr)

Item and Po are related with one to many relationships along with descriptive cost and quantity.

Supplier and Po are related with one-to-many relationships.

Create a RDB for the above and solve the following queries :

(ii) List the name of supplier to whom Po is given for "mouse".

→ Select S_name, I_qty from Supplier s, Item i , PO p

where s.S_no = p.S_no and p.I_no = i.I_no and I_name = "mouse"

8.Attempt the following :

Consider the following entities and relationships.

Item (I_no, I_name, I_qty)

Po (P_no, P_date)

Supplier (S_no, S_name, S_addr)

Item and Po are related with one to many relationships along with descriptive cost and quantity.

Supplier and Po are related with one-to-many relationships.

Create a RDB for the above and solve the following queries :

(iii) List the name of supplier and item_name in Po's generated on "30-sep-2009".

→ Select S_name, I_name from Item i, Supplier s, Po p

where s.S_no = p.S_no and p.I_no = i.I_no and to_date(P_date) = "30-sep-2009"

8.Attempt the following :

Consider the following entities and relationships.

Item (I_no, I_name, I_qty)

Po (P_no, P_date)

Supplier (S_no, S_name, S_addr)

Item and Po are related with one to many relationships along with descriptive cost and quantity.

Supplier and Po are related with one-to-many relationships.

Create a RDB for the above and solve the following queries :

(iv) List the names of suppliers who is going to supply “monitor” with minimum cost.

→ Select S_name from Supplier

where I_name = “monitor” and cost in(select min(cost) from po);

8.Attempt the following :

Consider the following entities and relationships.

Item (I_no, I_name, I_qty)

Po (P_no, P_date)

Supplier (S_no, S_name, S_addr)

Item and Po are related with one to many relationships along with descriptive cost and quantity.

Supplier and Po are related with one-to-many relationships.

Create a RDB for the above and solve the following queries :

(v) **Find out Po number, Po date and supplier name of the Po which is of maximum amount.**

→ Select P_no, P_date, S_name from PO p, Supplier s

Where p.S_no = s.S_no and cost in (select max(cost) from Po);

8.Attempt the following :

Consider the following entities and relationships.

Item (I_no, I_name, I_qty)

Po (P_no, P_date)

Supplier (S_no, S_name, S_addr)

Item and Po are related with one to many relationships along with descriptive cost and quantity.

Supplier and Po are related with one-to-many relationships.

Create a RDB for the above and solve the following queries :

(vi) Display all Po which contains the number, date, supplier name of the Po details of all items included in that i.e. name of item, qty and rate.

→ Select P_no , P_date , S_name , I_name, I_qty, cost from Po p, Supplier s, Item i

Where p.S_no = s.S_no and i.I_no = p.I_no;

THANK YOU!!!

My Blog : https://anandgharu.wordpress.com/

Email : gharu.anand@gmail.com