8/11/2021

MySQL is an opensource RDBMS

Data - Some infor or facts
Database - Collection of data
DBMS - software to manage data ex- excel
RDBMS - it is DBMS on which you can define relationship.

Table - Entity - a set of rows & columns.

RDBMS
- Support Relationship.
- Data stored in form of tables.
- Data can be accessed using language called as SQL (structured Query Language)
- Secured - Only users with permission can access your data.
- Supports transaction

MySQL Client.

Command line - i.e. Cmd to connect to mysql

mysql Workbench - GUI to connect to mysql

Default port for mysql - 3306

MYSQL - when we install mysql it installs instance of mysql.

→ System databases.
1) mysql
2) Information Schema
3) Performance Schema
4) Sys.

→ User Databases.

to connect.

- To command to change/ database:-
  use sakila;

- To list database ;-
  show database;

- To check which database is selected:-
  select database();

- To check tables in database:-
  show tables;

To delch table -
drop table t2

Default admin user in mysql is root.
i-e. Super user.

Table.
Table Name
Column Name.
Column Data type.

Create database DbVita; - To create database.
~~Create~~ use DbVita; - Use database.
Create table t1 (c1 int, c2 varchar(100), c3 date);

show tables; - To tables present in db.

To check content of table. (datatype).
- desc t1;

To show data from the table. (rows.
- Select * from t1;

To insert row in table.
- insert into t1 values (1, 'abc', '2021-01-01');

For specific column
Select c1, c2 from t1;

In select * from t1;
↑
is for all columns.

Pritom@123

insert into t1(c1,c2) values (5,'as');

to show specific rows.
- select * from t1 where c1=1;

Select * from t1 where c3 is null;

select * from t1 where c3 is not null;

Select * from t1 where c1<>1; not equal to

To exclud these rows with values
Selct * from t1 where c1 not in (1,4,5);

To show specific row with value :-
Selct * from t1 where c1 in (1,4,5);

To
Selct * from t1 where c2 like 'a%';

To show rows between the values.
select * from t1 where c1 between 1 and 4;

Sub queries

If we have two tables t1 & t2

Select *from t1 whm t1 in (select c1 from t2);
- to print rows ~~which~~ from t1 which contains same value of table t2.

- select *from t1 whm t1 > all (select c1 from t2);
to print valus from t1 which are greater than valus from t2

- Select * from t1, whm c1 > any (select c1 from t2)
to print vaulus from t1 and t2.

To show particulam rows:-
- Select * from t1 limit 3;

To updat row
update t1 set c2 = 'abc', c3 = '01-01-2021' whm
c1 = 3;

To delete row -
delete from t1 where c1 = '9';
delete from t1 whm c1 in (1,2,3);

To delete whole table:-
- delete from t1;

transaction - is used when it gives a option to commit and rollback the commands

Alter Commands:-

To add a column to a table
- alter table t1 add c4 int;

To drop column
- alter table t1 drop column c4;

To modify the datatype
- alter table t1 modify column c2 varchar(200);

To change the datatype to not null
- alter table t1 modify column c1 int null
- c1 int not null

To set default value:
alter table t1 modify column c1 int default 20;

To set default value at the time of creation of table.
Create table t4(id int default 1, name varchar

Constraints:-

Provides the some sort of restrictions to maintain integrity of data.

1) Unique key:- - it is a unique value from a table. - no unique value get duplicate.
- Single column can have multiple unique key.

Syntax:-
- Create table t5 (id int unique, name varchar(100));

2) Primary key - Can allow only unique values
- can not have null value
- multiple duplicate key not allows.
but does not have multiple primary key

Create table t6 (id int primary key, name(varchar(100));

To set primary key after the creation of the table.
- alter table t6 add primary key (id);

To drop primary key from a column.
- alter table t6 drop primy key;

Combination of two columns have primary key - it means c1 can have duplicates or can have dupliates but combination of both cannot have duplicates.

- Create table t7 (c1 int, c2 int, c3 (varchar 100)
primary key (c1, c2));

not null :-
Create table t8 (id int t8 (idint, iPint not null)

Check :-
Define what range will be allowed and not allowed

Create table t1 (id int, name varchar(100), chek (salary >= 0));

Create table t1 (id int, Gender varchar(100), check (gender in ('male', 'female'))

~~Parent~~ Foreign key - parent child relation

value is allowed in child only if it is in parent key
child table can have null values in ~~par~~ child key column.

Create table Parent1 (deptid int, primary key, deptname varchar (100));

Create table child (eid int, ename varchar (100), deptid int, foreign key fk1 (deptid) references parent1 (deptid));

To rename table.
alter table t1 rename to t6;

SQL
DDL - Data Definition Language - Defines the definition data not data
- Creak table
- Alter table
- Drop table
we can't rollback these command.

DML - Data Manipulation Language
Insert
Delete
Update
Merge - combination of update / delete / merge

We can rollback these commands.

DCL - Data Control Language.
Grant - give permisssions to user
Revoke - take permissions back from user.

TCL - Transaction Control language
Commit - Permanent changes.
Rollback - temporary changes ie. undo the changes.

DRL - Data Read language
Select - use to read data.

Aggregate function

- Count - works on all datatypes.
Count (*) - number of rows
Select count (*) from t1;

- Count (Col) - number of rows in specific columns which are not null.
Select count (c1) from t1;
Select count (c1), count (c2) from t1;

Select count (*) from t1 where c3 is null;
Select count (c1) from t1 where c3 is null;

sum

select sum(c1) from t1;
select sum(c1), sum(c2) from t1;

Avg

select Avg (c1) from t1;
select Avg (c1), Avg (c2) from t1;

min

select min(c1) from t1;
works with all datatype

max

select max(c1) from t1;
works with all data types.

groupby - is used to perform aggregation based on some columns.

select c2, sum(c1) from t1 group by c2;

select c2, sum(c1) from t1 when c2 = 'abc' groupby c2

when - to apply filters, can used on non aggregate col^n, without groupby.

having - aggregate columns. and with groupby.

on - clause to joining the condition.

Joins :- are used retrieve columns from multiple tables in the same query.

1) Cross join - Table with join without any condition.
- every row with every row.

Old syntax - select * from t1, t2
New Syntax - select * from t1 Cross join t2;

Inner join - with give those rows which satisfy the equality condition.
Equality condition T1.C1 = T2.C2

old syntax - Select * from t1, t2
when t1.c1 = t2.c1

new syntax :-
Select * from t1 inner join t2
on t1.c1 = t2.c1;

Select * from t1 join t2
on t1.c1 = t2.c1;

Outer join.

- Left outer join - will get all rows from left table and get matching row from right table

Syntax -

```
Select * from t1 left join t2
    on t1.c1 = t2.c1;
```

Right outer join - will get all rows from right table and get matching row from left table

Syntax

```
Select * from t1 Right join t2.
  on t1.c1 = t2.c1;
```

Full outer join

will get rows from both table and matching rows.

mysql does not support this.

Syntax - select * from t1 full join t2

```
on t1.c1 = t2.c1
```

non equality join:-

Select * from t1 join t2 where t1.c1 < t2.c1;
i.e there is no equality condition.

Set Operators:

- Union A Union B. (Need to perform sort to remove duplicates)
Select * from a
Union
select * from b;
- Union all - (Doesn't remove duplicates so no sort is required)
Select * from a.
Union all
Select * from b;
- Intersect (not supports in mysql)
(common elements)
Select * from a where id in (select id from b);

- minus / except - (not supported in mysql)
A minus B - ele which are not in B.
Select * from a where id not in (select id from b);

in and not in.

Pre-requisites for using set operators.

- Both the queries should have same no of Columns
- Data types of the columns in both should be same.

Having clause

only applicable when groupby method is used.

```
Select deptname, sum(salary) from emp join join dept
on emp.deptid = dept.deptid.
groupby deptname
  having Sum(salary) > 50000;
```

Order by clause.

To sort data in order

This is the last clause in sequence.

Column Alias

```
Select ename as EmployeeName , Salary as Employee Salary
    from emp;

select ename EmployeeName, Salary Employee Salary from emp;
```

Table Alias.

```
Select ename, salaray from emp e join deptd
on e.deptid = d.deptid;
```

limit clause.

```
select order by salary limit 1 offset 1;
```

1 → no of values

1 → ignoring one value

Inline view - when we put a query in from clause.

Giving a table alias for the query is mandatory.

```
Select * from (select ename, salary from emp
    order by salary desc limit 5) as t order by salary;
```

Analytical Function
Ranking fun^n - use to give Rank on rows with some condition.
used in 2 placed -
- select colum clause
- order by clause.

Two type of parameters.
partition by clause. - not mandatory.
order by clause - mandatory

row_number() - 1,2,3,4,5.
rank() - rank skips no in betn if we have dupliat value.

dense_rank() - number are not skipped if it have duplicate value duplicate rank will be given.

order by clause
ex - row_number over (order by salary desc)

partition by clause - ~~(row number()p.~~
(row number over (partition by deptid order by salary desc)

select * from (select deptname, ename, salary, row() number over (partition by deptname, order by salary desc) rn from emp join dept on emp deptid = dept.deptid) as t where rn<=5;

* Auto increment Column.
it take value automatically ~~se~~ in sequence.

Create table (accountno int primary key auto_increment, acctnaname varchar(100));

insert into ~~accou~~ (acctname) values ('abc');
- Here it will take accountno as 1 by automatically.

insert into (accountno, acctname) value ('101', 'xyz');
- Here it will take accountno and from next query it will automatically get increment.

alter table account auto_increment = 1001;

Case statements:-

Salarry >= 1000 - High >= 500 Medium anything else low.

syntax

```
Select *,
Case
when salary >= 1000 then 'High'
when Salary >= 500 then 'Medium'
else 'low'
end Salary_Band.
from emp;
```

Self join

join with itself.

Select * from emp e1 and emp e2

11/11/2021

View - a logical object, a saved query, a virtual table.

Doesn't store data.

Security Hide specific col^n & rows.

```
create view xyz
as
select c1, c2 from t1 join t2
on t1.a = b.a.
```

Insert into xyz value(1, 2, 3)

DML on Views

DML command will effect on base tables.

we can use delete, update, insert command on view then it will effect on base tables.

DML command will not applied on multiple tables.
Delete on a view will not work on joined table

Restrictions - 1) If the view is of joined two tables then we can't insert data on both table at a time using view

2) If a table contains two columns where one column is not null and view contains other column then we can not insert data.

- delete can not perform from view on join.
- update ——————— because group by command.

i.e. when we have any aggregation then cannot perform view ?

Codd Rules

Rule Zero :-
For a system to qualify as RDBMS then able to manage databases through / by relational capabilities.

1) Information Rule.
- All info has to be stored in tables.
- the rows & columns have to be unordered

2) Guaranteed Access.
each unique bpiece of data should be accesible by → table name + primary key (row) + column

3) Systematic treatment of NULL
- Null means - Missing data, Not applicable, No value.
- primary key - not null
- expression on Null should return Null
- null should be displayed as Null only. not as zero.

4) Active online Catalog.

- Database dictionary (catalogs) to have discription of the database

- Database catalog should be stored in tables.

- Same queries should to be used on the catalog database as on application database.

5) Powerful language.

- a language to provide all manners of access to the data

  ex - SQL

- If file supporting table can be accessible by any manner except a SQL interface then a violation.

6) View Updation Rule.

- View - Virtual table temporarily derived from table.

- not updable - View does not have Non-Null column of base table. then it cannot be updatable.

- updation on View does not apply on computed fields.

7) Relational level operations.
- There must be insert, update, delete operations of the level Relations (table)

- set operations like Union, Intersection & minus should be supported.

8) Physical Data Independence.
- physical data storage should not matter to the system.
- If file supporting table etable was renamed moved from one disk to another it should not effect the applications performed on data.

9) Logical Data Independence
- If there is change in the logical structure of the database the user view of the data should not change.

- Difficult one to satisfy.

- implemented through views. if a table is split into two tables a new view should give a result as the join of the two tables.

10) Integrity Independence -

- Database should be able to use its own integrity rather than using other language.
- integrity rule / (keys & check constraints)
  - filter to allow correct data, should be stored in Data Dictionary.
- key & check constraints, triggers etc should be stored in Data Dictionary.

11) Distribution Independence

- A database should work properly regardless of its distribution across the network.
- This lays foundation of Distributed Database.

12) Non subversion Rule

- If a user have access to low level data, then it should not be able to bypass integrity rules to change data.
- This may be achieved by some sort of locking encryption.

Correlated Subquery -

Subqueries in which we define a relation of a column from a outer query with the column of inner query.

- Subquery gets executed as many times as you have number of rows in the outer query.

Syntax

```
select ... from out_query.
where (some inner query where outer query.
Column = inner query.column)
```

Exists & not exists - operators.

- you write a query after those operators.

- If query returns 1 or more than 1 row then the condition becomes true otherwise is set to false.

- Generally are used in context with a correlated subquery.

```
Select * from emp where exists (select *
d.deptid from emp e join dept d
on e.deptid = de.deptid);
```

Show Variables like 'autocommit'

Truncate vs Delete

1) Truncate does not have where clause.
so truncate delete all data from table
delete will delete specific data / rows

2) Truncate cannot be rollback.

3) logging data is done with delete command
but logging data is not done with truncate command.

logging data takes time

4) so truncate is faster than ~~be~~ delete.

5) truncate resets the auto increment value
but delete doesn't reset the auto increment.

Can
6) Create a trigger on delete
but cannot create command on truncate.

Transeaction - Commit, Rollback.

Properties - A - atomicity C - consistency I - Isolation
D - Durability.

A - Transaction is either fully ~~comma~~ a commit or rollback.
- transaction should be treated as atomic unit.

C - Database should always remain in consistent stat after any trasaction irrespective of whether transaction is commited or rollback or not completed.

- RDBMS writes the changes first to log file before to change the changing the data in the buffer pool or data file this is called write Ahead logging.

- Consisteny achieved by Instance Recovery, done during startup of instance.
- Redo all the transaction that were commited but not written to the disk
- Undo or rollback all the transaction which were not commited but got written to the disk
- Consistency implement by logg

I - Isolation - No two users can update the same data at the time RDBMS use locks to implement isolation

- MVCC - multi version concurrency control
- It means that user can read data even

if the same data is getting modified by some other session/user.

- DML command takes exclusive lock.
- Select command takes share lock.

Isolation implemented by locks

- Durablility - long lasting
- Once database stored in RDBMS it should remain for ever even if the server is restarted unless the user deletes the data.

- Durability is implemented by storing the data on non-volatile storage.

show variables like

Select substring ('abcdef', 1, 3); — abc
Select substring ('abcdef', -2, 2); — ef.

Select Instr ('abcdef', 'de') - number of de - 4.

Select Instr ('abcdef', 'f', 'k') — abcdek

Select left ('abcde', 3) — abc

Select Right ('abcde', 3) - cde

Select ltrim ('___kk___') — kk___
Select rtrim ('___kk___') — ___kk
} Remove space

Select trim ('___kk___') — kk

Select replace ('abc de', ' ', ''); —> abcde.

Select length ('abcd'); ⟶ 4

Select upper ('abcd'); ⟶ ABCD

Check constraints after 8 version.

STUDENT'S ROLL NO.

DATE ON WHICH EXPT. WAS PERFORMED

M. K. S. S. S.
CUMMINS COLLEGE OF ENGINEERING FOR WOMEN
PUNE 52.

Date funⁿ

```
select current_date();   } current date
select curdate();

select current_time();   } current time.
select curtime();

select now();  - current date & time.

select year(now());  - current year
select month(now());  - current month
select day(now());  - current day
select hour(now());  - current hour
select minute(now());  - current minute
select second(now());  - current second
select weekday(now());  - current day from week
select week(now());  - current no of week from year
select last_day(now());  - last day
select datediff('2021-12-01', now());  - date difference
select date_add('2021-12-01', interval 2 day);
                 - add 2 days with interval of two
       ('2021-12-01', interval 2 month);
                 - add days with interval of two months

select date_format(odate, '%Y_%M') from orders
```

odate ↑ column name.
'%Y' ↑ capital for whole year
'%M' ↑ capital for month name
orders ↑ table name

(17)

What is Isolation level?

- It defines what the select query will return if the same data is getting updated in another session.

1) Read uncommited
2) Read commited
3) Repetable Reads - Default isolation level
4) Serializable.

Blocking -
One session is waiting for the another session to commit / roll back / for the query to finish. So that the lock is released.

1) Read Uncommited:-
- To ~~read~~able to read uncommitted data or dirty data
- Uncommited data - data which has been modified but ~~yet~~ not yet Commited / rollback.

To chek isolation level
-show variables like '%. isolation%';
~~Default~~
To chang isolation level
set trasaction_isolation = 'read-uncommited';

cmd-1
set transaction_isolation = 'read-uncommitted';
Stp-1 · start transaction
update emp set salary = 200 where eid=1;

Cmd-2
set transaction_isolation = 'read-uncommited';
Stp-2 - Start trasaction
select * from emp where eid=1
(it will return 200 as salary.
update emp set salary = 300 where eid=1;
→ waiting.

Repeatable

2) ~~Readable~~ Read
Reading the same data in same transaction even if the data is committed or not

3) Read Commited:-
- In read commited isolation Select query will be able to read last committed data
- Repeatable reads not happening.

cmd1 set transaction_isolation = 'read-commited'
Start trascaction
update emp set salary =200 where eid=1;

cmd2 ~~Sta~~ set transaction_isolation = 'read-committed'
start transaction
select * from emp where eid=1 ← 1000

it will not return 200 until it gets commited.

4) Repeatable read;

Reading the same data in a transaction even if the data has been modified & commited in another session

Select query transaction if we insert the data and commit then it will reflect in first then it will not reflect in first session.

my sql does not phantom reads:- ghostrecords
- if a transaction stay in first and in second session the data is updated and still it shows in first transaction also

4) Serializable :-
Phantom read.
In Select query, if we start transaction & if we insert data in second session & commit it. then it will reflect in first transaction session in same transaction.

ROLL NO. _____

HICH EXPT. WAS PERFORMED _____

CUMMINS COLLEGE OF ENGINEERING FOR WOMEN
PUNE 52.

If deadlock happen often mysql will rollback a ~~Locks~~ and release a lock from a from table.

Procedures.

- Set of code/programs where we can take i/p through parameter and can also return values from the parameters
- Code procedure cannot be used in query

delimiter - to change command to end.

Functions:-

Set of code which take i/p through some parameter but has to return a value through return clause. You can call fun^n inside queries.

Create fun^n name (P1 datatype)
Returns datatype.
Deterministic
Begin
Delon

set var
if
else
endif

end loop;
return statement;
end.

Types of fun^n

Deterministic -
if the fun^n is expected to return same output for the same input value each time.

Read SQL Data:-
if the function is having select from table Comamand

No SQL - ↓
If the fun^n doesn't read data from SQL table.

Cursors:-
is a pointer to a set of records and is used fetch data row by row.

Declare cursor
Declare not found handler for cursor.
Open cursor.
fetch data from cursor in a loop.
Close cursor.

OLTP VS OLAP

OLAP - Online Analytic Procedure.
OLTP - Online Transactional Processing.

OLTP -
- Database is optimized for transaction - DML commands
- OLTP databases are highly normalized maybe upto 4th or 5th NF

OLAP -
- Database is optimized for mainly select commands.
- They are only normalized upto 2nd or 3rd NF

Types of OLAP:-

Dimensions:-

A dimension table consist of dimensions of the fact.

Data in dimension table doesn't change very frequently.

Facts / measures:-

- Actual facts about your dimension.
- Data in the fact table changes very frequently.

OLAP Data warehouse Data models.

Star schema:- 2NF

You will have dimension & fact table. The bet^n a dimension table & fact table, it means you cannot have relationship bet^n any two facts & two dimensions.

Snowflake Schema - 3NF

You will have dimension & fact table. The relationship can exists bet^n dimension table & fact table and bet^n any two dimension but not bet^n any two facts.

B- Tree Indexes

* Clustered index :-
  - Entire row of the table is stored in the leaf level of the index.
  - entire row is stored in the leaf you cannot create more than 1 clustered index on a table.

  - In mysql for Innodb table when you create a primary key a unique clustered index is created on the table automatically

  - when a clustered then internal structure of the table is dropped.

* Non- Clustered Index :-
  Only the ROWID is store along with index column in the leaf.
  Table can have multiple non clustered indexes.
  when ever you create a unique key in the table a unique non- clustered index is added automatically.

Triggers:-

Set of code that gets executed whenever a specific event occurs Trigger is like a hidden code.

events -

DML Commands.
Insert, update, delete.

Logon Triggers
Login on mysql

System Commands.
Startup, shutdown of mysql services.

DDL Commands.
Create, Drop, Alter

Triggers are generally used for auditing purpose & automation purpose.

Indexes:- are objects which are used to optimize search.

1) ROWID - 1
Physical address of each row. it is available to read in Oracle but not in mysql

2) ROWNUM -
It is a sequential number assigned to result set.

Engines:-

InnoDB :- Default engines
myISAM :- Default engine prior versions:1
CSV
Archive
memory

INNODB:-
- Supports transaction
- Fully ACID compliants
- Foreign key
- Log file and Data file
- Clustered index
- DML
-

MyISAM :-
Create table t_myisam (id int) engine = myisam

- Can only perform DML
- not all properties of InnoDB

CSV :-
Create table t_csv (id int not null, name varchar(100) not null engine csv;

- Has data in csv format.
- all columns should be defined as not null for csv engine
- Can perform DML

Archive :-
Create table t_archive(id int, name(varchar(100)) engine = archive;
- Has Data file
- ~~Perform~~ DML Perform insert and select
- Delete update, truncate not allowed.

memory
- Can perform insert, update, delete

declare i int;
Set @constant - To declare constant

open loop, for loop, while loop

explicit cursor - define using declare cursor.
implicit cursor - cursor created by mysql internally to execute some queries.

Declare a handle.
- Declare action

materilised view stores data & does occupy space for the data

Savepoint :- rollback to specific command.

Source c:\backup\sqlcmd.txt.