

# Databases



***"Quality of an application largely depends on the way the data is stored  
and***

***Future of a business depends on the way the stored data is used"***

- Prafull, Founder & Technology Head



MICROSOFT PARTNER | ISO 9001:2015

**SOHAMGLOBAL**

Training | Projects | Internships



## { SohamGlobal }

We are an ISO 9001:2015 certified, Microsoft Partner software development and training consultancy. We are involved in activities like commercial application development, training & certification on enterprise class technologies, academic projects development, media production, authoring technical books, conducting new technology workshops etc.

We give you tools to be able to explore the universe. We believe - technology is power; technology is future. Technology has revolutionized the human life more than anything else. We have been promoting and using advanced technologies for last 20 years. Along with our clients, international giants like Microsoft & Oracle have appreciated our sincere efforts in training and development on technologies like JavaEE, Microsoft.NET, Python, Cloud, BigData, Data Analytics etc. Our constant industry research & services have put us at a very respected position. We do not consider anything more important than customer satisfaction. We have a team with the right expertise and attitude to deliver our objectives. We also work for upliftment minorities & dalits thru technology education, child rights, Paani foundation and promote eco-friendly learning thru our "Green Students" initiative.



**Prafull**

Founder & Technology Head



**Sharayu**

Research & Training Head



**Megha**

Projects Coordinator

*World class team; world class services.*

# Introduction to databases:

---

**Database** is a place where data is stored systematically and managed using some predefined tools. Programming technologies like Java consult this place to get data for processing and generate information for their users. Database is the biggest support that an application can have. Almost every activity a program performs in a business application needs database assistance. Companies' information, transactions, decisions, productivity, intelligence and growth largely depends on the quality of database they have.

New domains of development like Big data, Data Science, Data Analytics, Machine learning, AI, etc are all data oriented.

A database is an organized collection of data, so that it can be easily accessed and managed. We can organize data into tables, rows, columns, and index it to make it easier to find relevant information.

Modern databases are managed by the database management system (DBMS) or RDBMS.

**SQL or Structured Query Language** is used to operate on the data stored in a database. It is common language that all the databases use to store, manage and retrieve data. Programs also use SQL to talk to the database data.

SQL instructions or queries are usually categorized as

- DDL or Data Definition Language
- DML or Data Manipulation Language
- DCL or Data Control Language

**Data Definition Language:** DDL works on objects. Operations like creating objects, modifying object definitions and removing unwanted objects from the DB fall under DDL. Query examples are Create, Alter, Drop, etc.

**Data Manipulation Language:** DML works on data. Operations like inserting new data in the objects, modifying data, removing data and generating information from the data store are examples of DML. Query examples are Insert, Update, Delete, Select, etc.

**Data Control Language:** DCL works on transactions. Maintaining consistency of transactions by confirming or cancelling client side data operations on the DB server is done by DCL. Failure in DCL can result into serious data irregularities. Query examples are Commit, Rollback, etc.

There are databases of different sizes and applicability. Databases can be classified into –

- Small scale databases
- Mid range databases
- Large scale / Enterprise databases

Databases can be structured or unstructured. Structured databases use a predefined structure to store data whereas unstructured databases can store data in any random format.

Oracle, Microsoft SQL Server, DB2, MySQL etc. are some of the most popularly used databases.

**Normalization:** Database normalization is the process of structuring a relational database in accordance with a series of so-called normal forms in order to reduce data redundancy and improve data integrity. It makes sure that all data is not stored at one place, it is separated and stored at different places.

Normalization reorganizes data in a database so that it meets two basic requirements: (1) There is no redundancy of data and (2) data dependencies are logical (all related data items are stored together). Normalization is important for many reasons, but chiefly because it allows databases to take up as little disk space as possible, resulting in increased performance.

**Entity Relationship Diagrams (ERD):** An entity-relationship diagram (ERD) is a data modelling technique that graphically illustrates an information system's entities and the relationships between those entities. An ERD is a conceptual and representational model of data used to represent the entity framework infrastructure.

Technically speaking a database is a collection of objects that are used to store and manage data. These objects are used in application development while working with the database in programming. Java has JDBC and .NET has ADO.NET for database programming.

Database programming requires solid knowledge of –

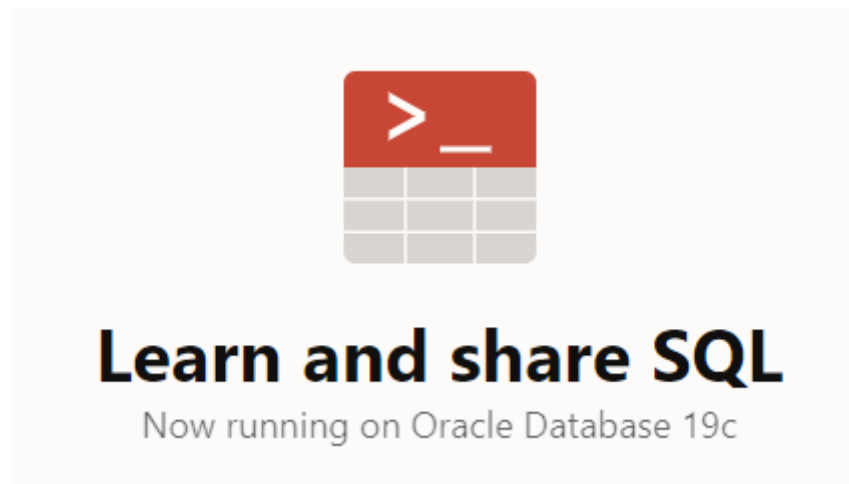
- Basic queries
- Constraints
- Views
- Stored Procedures
- Triggers
- Joins
- Index
- Functions

Queries for

# MySQL8

# &

# LiveSQL Oracle 19c



*Learn Oracle, worlds #1 database with SohamGlobal*

# Various databases

---

There are various databases available in the market developed by different companies and communities. We can choose a database according to our requirements and its style of data storage.

## Structured SQL relational databases:

- Oracle
- MySQL
- Microsoft SQL Server



It is a database commonly used for running online transaction processing (OLTP), data warehousing (DW) and mixed (OLTP & DW) database workloads. The latest generation, Oracle Database 19c, is available on-premise as well as on-cloud

Gartner Recognizes Oracle as a Magic Quadrant Leader in Database Management

Oracle's revolutionary cloud database is self-driving, self-securing, self-repairing, and designed to eliminate error-prone manual data management.



MySQL is world's most popular free open-source relational database management system. Its name is a combination of "My", the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language. MySQL is one of the best RDBMS being used for developing various web-based software applications. It is supported by an active

community of open source developers and enthusiasts.

Originally developed by MySQL AB, then bought by Sun Microsystems and now owned by Oracle Corp.

It is used by Facebook, Twitter, Youtube, etc.



Microsoft SQL Server is a relational database management system developed by Microsoft. As a database server, it is a software product with the primary function of storing and retrieving data as requested by other software applications.

Like Oracle it is also used on premise as well as a cloud service

It is scalable for small, medium and enterprise class applications

## Unstructured NoSQL databases

- MongoDB
- Cassandra



MongoDB is a cross-platform document-oriented database program.

Classified as a NoSQL database program, MongoDB uses JSON-like documents with schema. MongoDB is developed by MongoDB Inc.

Example-

```
{ "name": "ethan hunt" , "company": "sohamglobal" }
```



Apache Cassandra is a free and open-source, distributed, wide column store, NoSQL database management system designed to handle large amounts of data across many commodity servers, providing high availability with no single point of failure.

Cassandra offers robust support for clusters spanning multiple datacenters, with asynchronous masterless replication allowing low latency operations for all clients.

This database is written in Java

**This PDF & associated video series describe SQL queries for structured databases only.  
Download separate PDF for MongoDB and Cassandra NoSQL unstructured databases**

# SohamGlobal

*recommends*



*World's #1 most powerful popular  
enterprise database.*

Use it online for free from the cloud.

**[livesql.oracle.com](https://livesql.oracle.com)**

*Best database for your future.*



# Basic Queries

---

The instructions that we give to the database are called as Queries.

```
create table accounts
(
  accno int primary key,
  accnm varchar(30),
  acctype varchar(10) not null,
  balance float check(balance>=500)
);
```

**Note: primary key, NOT NULL and check are constraints that are explained in the next section of this document.**

```
insert into accounts values(1001,'sachin tendulkar','saving',45000);
insert into accounts values(1002,'maria sharapova','fixed',23100);
insert into accounts values(1003,'mitchell johnson','current',15300);
insert into accounts values(1004,'edan hazard','saving',41300);
insert into accounts values(1005,'robert lewandowski','saving',21600);
insert into accounts values(1006,'gabriel jesus','fixed',37100);
insert into accounts values(1008,'marcos alonso','saving',33900);
insert into accounts values(1009,'david luiz','current',28400);
insert into accounts values(1010,'glenn maxwell','fixed',61150);
insert into accounts values(1012,'zaheer khan','saving',22570);
insert into accounts values(1013,'antoine griezman','fixed',37640);
insert into accounts values(1014,'mesut ozil','saving',53760);
insert into accounts values(1015,'paul pogba','saving',67370);
insert into accounts values(1016,'garry cahill','current',39550);
insert into accounts values(1017,'thibaut courtois','fixed',97550);
insert into accounts values(1018,'cesar azpilicueta','current',82716);

insert into accounts values(1019,'cesc fabregas','current',82716);
insert into accounts values(1020,'ngolo kante','fixed',55871);
insert into accounts values(1021,'victor moises','current',63228);
insert into accounts values(1022,'willian','saving',75961);
insert into accounts values(1023,'pedro','current',26551);
insert into accounts values(1024,'michy batshuayi','fixed',63558);
```

```
insert into accounts values(1025,'andreas christensen','saving',68523);
insert into accounts values(1026,'alvaro morata','saving',96995);
insert into accounts values(1027,'davide zappacosta','current',49668);
insert into accounts values(1028,'daniel drinkwater','fixed',75442);
insert into accounts values(1029,'mohamed salah','saving',83229);
insert into accounts values(1030,'sadio mane','current',45889);
insert into accounts values(1031,'roberto firmino','fixed',96321);
insert into accounts values(1032,'alex oxlade chamberlain','current',58449);
insert into accounts values(1033,'alexandre lacazette','saving',36227);
insert into accounts values(1034,'raheem sterling','fixed',62994);
```

```
insert into accounts values(1035,'harry kane','current',78751);
insert into accounts values(1036,'dele alli','fixed',96637);
insert into accounts values(1037,'hugo lloris','saving',84552);
insert into accounts values(1038,'bill gates','current',93119);
insert into accounts values(1039,'amir khan','saving',66554);
```

```
insert into accounts(accno,acctype,balance) values(1045,'saving',13900);
```

**Note: Copy all insert queries into your database table for further practice.**

---

```
Update accounts
set acctype='fixed'
where accno=1003;
```

```
Update accounts
set balance=balance+(balance*3/100)
where acctype='saving';
```

```
Update accounts
Set balance=balance-150;
```

---

```
Delete from accounts where balance<1000;
Delete from accounts;
```

---

**Select \* from accounts;**

Select accnm,balance from accounts;

Select \* from accounts

where acctype='saving' **and** balance>50000;

Select \* from accounts

where acctype='fixed' **or** balance>100000;

Select \* from accounts

where balance **between** 20000 **and** 45000;

Select accnm from accounts

where acctype **in** ('saving','fixed','current');

Select \* from account

where **not** acctype='current';

Select \* from accounts where accnm **like** 's%';

Select \* from accounts where accnm like '\_a%';

Select \* from accounts **order by** accnm;

Select \* from accounts order by balance desc;

**Aggregate functions-**

**sum(),avg(),min(),max(),count()**

Select avg(balance) from accounts

where acctype='fixed';

Select count(accnm),sum(balance) from accounts;

Select count(accnm) 'total acc', avg(balance)

'average bal' from accounts

where acctype='saving'

Select acctype,count(accnm),sum(balance)

from accounts **group by** acctype;

Select \* from accounts  
Where balance > (select avg(balance) from accounts);

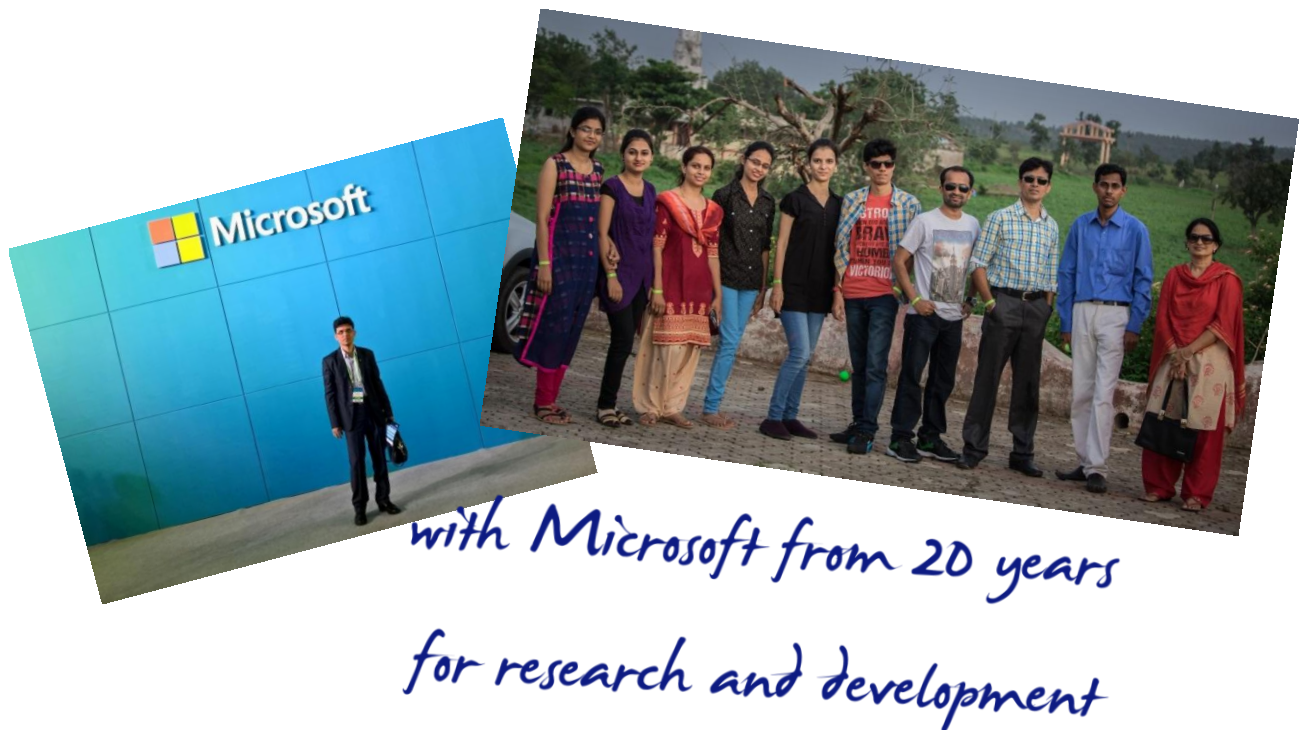
Select accnm from accounts  
Where balance=(select max(balance) from accounts);

Select \* from accounts  
Where acctype=(select acctype from accounts where accnm='adam gilchrist');  
alter table doctors add mobile varchar(10) unique;

alter table patients modify patnm varchar(50);

Drop table students;

Queries in this PDF are explained in detail with more options in the video tutorial series.





# Constraints

---

**Constraints are the rules and restrictions applied on the table data to ensure consistency and integrity of data across the system. The programming technologies don't need to write code for the same.**

Need:

- Every business has some standards for data storage
- Constraints are the rules and restrictions applied on the table data to ensure integrity and consistency across the system
- We can make sure that no bad, unacceptable data is stored in the columns of a table

Examples of restrictions:

- Two customers should not have the same customer ID
- Age of an employee must be more than 21 years
- Every student should have a separate mobile number & email id
- Author of a book can not be empty
- Station code available in the list of stations can only be allotted to passengers for reservation

Constraints:

- Primary key
- Unique
- Not null
- Default
- Check
- Foreign key

## Primary Key:

- It is used to uniquely identify a row/record in a table.
- It can not be repeated
- It can not be left blank

Example-

```
> Create table trains (trainno int primary key,  
trainnm varchar(30) , fromstation varchar(20), tostation varchar(20));
```

## Unique

- It does not allow a column value to repeat.
- Ensures all values in the column are different

Example-

```
>Create table students(rollno int primary key,  
studnm varchar(30), mobile varchar(30) unique);
```

## Not Null

- This constraint ensures that the column values can not be NULL or empty

Example –

```
Create table book(bookid varchar(5),booknm varchar(50),  
author varchar(20) Not Null);
```

## Default:

- Specifies a default value to a column when no value is supplied from the insert query.

Example-

```
Create table employee (empno int primary key,empnm varchar(30),  
dept varchar(15) default 'marketing');
```

## Check:

- It ensures that all values in the column satisfy the given condition

Example-

```
Create table customers(custno int primary key, custnm varchar(30),age int check(age>=18));
```

### Foreign Key:

- It is used to establish relationship between data of one table with the data on another
- Every activity like insert, update or delete is performed only after making sure that the reference rules of the two tables are maintained
- Only values present in the primary key table can be used in the foreign key, new values are not allowed
- Row in the primary key table can't be deleted if there is a reference of that data in other table

Doctors table

DoctorID-PK	DoctorNm	Specialization
D7145	Morgan	Cardiology
D0913	Buttler	Gen surgeon
D2601	Anderson	Orthopaedic
D1186	Gilchrist	Neurology

Patients table

PatientNo	PatientNm	TreatedBy-FK
101	Donald	D0913
102	Boris	D1186
103	John	D1186
104	Ricky	D7145
105	Andrew	D8299

This row will be rejected as D8299 doesn't exist in Doctors table



**Rule 1: TreatedBy column of Patients table can accept only the values that exists in DoctorID column of doctors table**

**Rule 2: Only record of D2601 can be deleted from Doctors table but others can't as they have reference in the patients table**

**Conclusion:**

- Constraints must be compulsorily used while creating tables as they maintain quality, integrity and consistency of data
- Tables without constraints put a lot of burden on the programming technologies for maintaining restrictions on data



*Workshops & presentations in colleges  
on BigData, Blockchain, etc.*



```
create table emp
(
empno int primary key,
empnm varchar(30),
dept varchar(15) not null,
mobile varchar(10) unique,
qualification varchar(10) default 'mumbai',
age int check(age>=18),
salary float
);
```

```
create table customers
(
custno int primary key auto_increment,
custnm varchar(30),
mobile varchar(15)
);
```

```
create table CountryMaster
(
CountryID int primary key,
CountryName varchar(100)
);
```

```
create table StateMaster
(
StateID int primary key,
CountryID int,statename varchar(100),
FOREIGN KEY(CountryID) REFERENCES CountryMaster(CountryID)
);
```

```
create table LocationMaster
(
LocationID int primary key,
StateID int,LocationName varchar(1000),
IsSubLocation bit,
FOREIGN KEY(StateID) REFERENCES StateMaster(StateID)
);
```

---

```
create table flights
(
  flno int primary key,
  flfrom varchar(20),
  flto varchar(20),
  company varchar(20),
  fldttm datetime not null
);
```

```
insert into flights values(1422,'mumbai','london','etihad','2019-05-21 8:30');
insert into flights values(1951,'new delhi','chicago','air india','2019-03-05 11:45');
```

```
create table traveller
(
  custno int primary key auto_increment,
  custnm varchar(30),
  flight int,
  people int,
  ticketamt float,
  foreign key fl (flight) references flights(flno)
);
```

```
insert into traveler(custnm,flight,people,ticketamt) values('praffull',1532,1,23000);
```

```
ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint fails
(`sharayudb`.`traveler`, CONSTRAINT `fl` FOREIGN KEY (`flight`) REFERENCES `flights` (`flno`))
```

```
insert into traveler(custnm,flight,people,ticketamt) values('praffull',1951,1,23000);
```

# Views

---

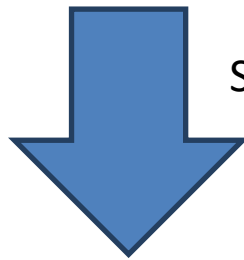
**View is a logical or virtual table that stores the most required data from large tables resulted by a select query. This improves performance of data access. Every change made to the table is automatically reflected in the view.**



- It is a logical table that stored selected data from database tables
- It improves performance of data access by reducing efforts and time of data retrieval
- Unlike ordinary base tables in a relational database, a view does not form part of the physical schema
- It is a virtual table computed or collated dynamically from data in the database when access to that view is requested
- Views don't waste disc space
- Views can hide the complexity of data access.
- Changes applied to the data in a relevant underlying table are reflected in the view automatically
- Select the most required data from table and put it in a view
- Programs will access the data from the view quickly

## One or more Tables with lots of data

RollNo	StudentNm	Branch	Semester	Mobile	...
3526	Mohammed Salah	Computers	6	273688273	...
6472	Sadio Mane	Electronics	8	637848832	...
4738	Virgil Van Dijk	Mechanical	8	26384899	...
:	:	:	:	:	:



Select Query

## View with only required selected data

Name	Mobile
Mohammed Salah	273688273
Edan Hazard	492788294
Kepa Arizabalaga	847366273
:	:



Programs and Applications

### Conclusion:

- We can create views for the most frequently required data and use it in programming applications for better performance

```

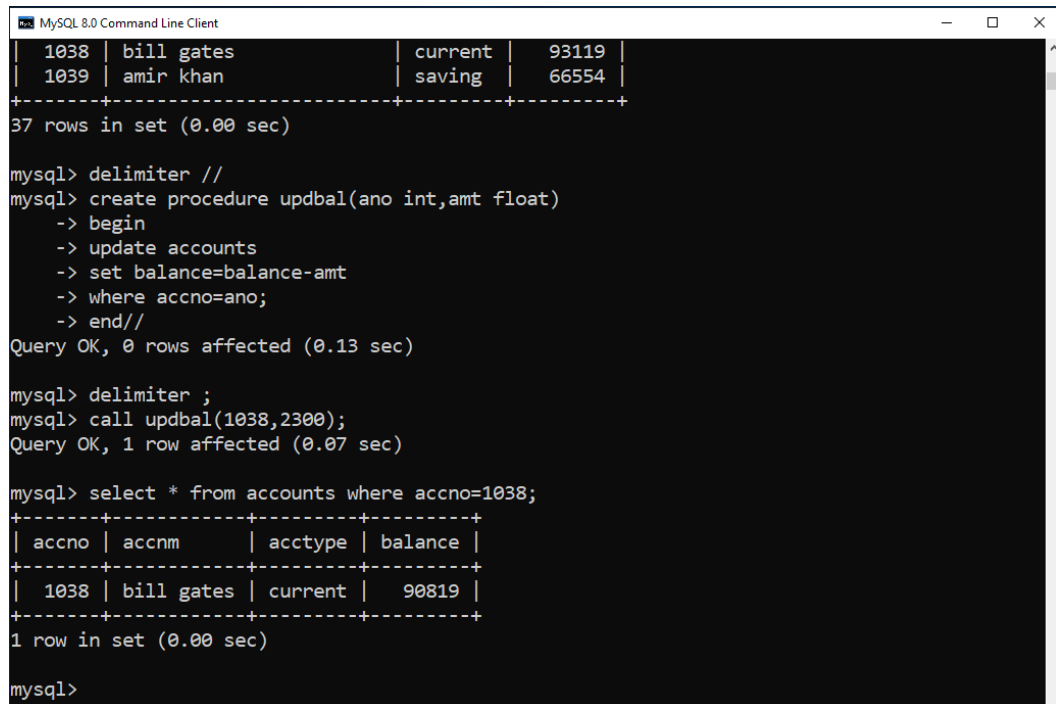
create view v1
as
select accnm 'Name',
balance ,
balance*3.5/100 'Interest'
from accounts
where acctype='saving';

```

```
select * from v1;
```

```
delete from v1 where Name='Zaheer';
```

```
update v1 set balance=balance+333 where Name='soham';
```



```

MySQL 8.0 Command Line Client
+----+-----+-----+-----+
| 1038 | bill gates | current | 93119 |
| 1039 | amir khan | saving | 66554 |
+-----+-----+-----+-----+
37 rows in set (0.00 sec)

mysql> delimiter //
mysql> create procedure updbal(ano int,amt float)
-> begin
-> update accounts
-> set balance=balance-amt
-> where accno=ano;
-> end//
Query OK, 0 rows affected (0.13 sec)

mysql> delimiter ;
mysql> call updbal(1038,2300);
Query OK, 1 row affected (0.07 sec)

mysql> select * from accounts where accno=1038;
+-----+-----+-----+-----+
| accno | accnm | acctype | balance |
+-----+-----+-----+-----+
| 1038 | bill gates | current | 90819 |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql>

```

*FREE work from home*

*Internship program*



*and global certifications...*

**We offer training, certifications and internships on world's  
finest technologies**

# Stored Procedure (PL/SQL block in Oracle)

---

A procedure is a collection of SQL statements that can be called with a name.

A stored procedure improves security, integrity and productivity of a programming application. It reduces complexity of code by avoiding programming of database transactions in a program. Use of procedures is highly recommended in a project.

- It is a collection of SQL statements that can be saved and reused to perform operations on database data
- It is a code that can be used by programs to perform activities without writing long SQL queries
- It reduces the complexity of code and improve security, efficiency and performance of database operations
- Series of SQL statements can be executed by a single instruction
- SP help in developing software applications
- Programming technologies like JDBC provide special classes like CallableStatement to execute Stored Procedures

## Types of stored procedures

- Parameterized stored procedures need data from the programs to perform database activities
- Non parameterized stored procedures don't need any data from outside to perform operations



### Example:

```
Create Procedure transferamt(sno int,dno int,amt float)
AS
BEGIN
Update accounts set balance=balance-amt
where accno=sno;
Update accounts set balance=balance+amt
where accno=dno;
Insert into transferlog values(.....);
END
```



Call transferamt(1062,2859,1500);

### Conclusion:

- Use of stored procedures is highly recommended as it improves overall quality of a software application

Creating a stored procedure

**DELIMITER //**

```
CREATE PROCEDURE updbal()
BEGIN
update accounts set balance=balance+(balance*3/100)
where acctype='saving';

update accounts set balance=balance+(balance*7/100)
where acctype='fixed';

update accounts set balance=balance+(balance*4/100)
where acctype='current';
END//
```

**DELIMITER ;**

Executing procedure

**call updbal;**

---

**DELIMITER //**

**CREATE PROCEDURE newacc(ano int,anm varchar(30),atyp varchar(10),bal float)**

**BEGIN**

**insert into accounts values(ano,anm,atyp,bal);**

**END//**

**DELIMITER ;**

**call newacc(1007,'jos butler','saving',36500);**

**delimiter //**

**create procedure addbonus(ano int,amt float)**

**begin**

**update accounts set balance=balance+amt**

**where accno=ano;**

**end//**

**delimiter ;**

**call addbonus(1001,500);**

---

**NOTE: DELIMITER is used to enclose multiple queries in one definition on the MySQL command line client**

### Procedure that transfers amount from one account to another

DELIMITER //

```
CREATE PROCEDURE trans1 (sacc int,dacc int,amt float)
BEGIN
declare tot float;
select balance-amt into tot from accounts where accno=sacc;

if tot>=0 then
update accounts set balance=balance-amt where accno=sacc;
update accounts set balance=balance+amt where accno=dacc;
end if;
END//
```

DELIMITER ;

call trans(1017,1013,5500);



delimiter //

```
create procedure trans(no int,typ varchar(10),amt float)
begin
if typ='deposit' then
update accounts set balance=balance+amt where accno=no;
else
update accounts set balance=balance-amt where accno=no;
end if;
end//
```

delimiter ;

call trans(1002,'deposit',5300);

---

delimiter //

```
create procedure getname(no int)
begin
select stnm from students
where rollno=no;
end//
```

delimiter ;

```
call getname(102);
```

---

DELIMITER //

```
CREATE PROCEDURE GetAllProducts()
BEGIN
SELECT * FROM products;
END//
```

DELIMITER ;

## Procedures in Oracle 19c LiveSQL

```
create or replace procedure updbal(ano IN int,amt IN int)
AS
BEGIN
update accounts
set balance=balance-amt
where accno=ano;
END;
```

```
call updbal(1039,1000);
```

```
create or replace procedure getname(ano IN int)
IS
anm accounts.accnm%TYPE;
begin
select accnm into anm from accounts
where accno=ano;
DBMS_OUTPUT.PUT_LINE('Name: ' || anm);
end;

call getname(1001);
```

## *SohamGlobal works with Paani Foundation*



# Cloud database

---

## What is cloud based database?

**Cloud database means a database that is accessible to clients from the cloud and delivered to users on demand via the Internet from a provider's servers. Also referred to as Database-as-a-Service (DBaaS), cloud databases can use cloud computing to achieve optimized scaling, high availability, multi-tenancy and effective resource allocation.**

A cloud database is a collection of data and information content, either structured or unstructured, that resides on a private, public or hybrid cloud computing infrastructure platform. From a structural and design perspective, a cloud database is no different than one that operates on a business's own on-premises servers. It only differs from the place it is delivered to the application and end users.

Cloud Database service providers:

- Microsoft Azure SQL Database
- Amazon Relational Database
- Oracle Database
- IBM Db2 on Cloud
- Google Cloud SQL



## Amazon:

Amazon Relational Database is a Database as a Service (DBaaS). It is suitable for experienced data users, data scientists and database administrators. Users need to contact a Database Administrator to get setup as the process is technically involved. Users can build databases specifically geared around their needs. You can create templates or write code. Users can control the type of database, as well as where data is stored. Specific database formats that are supported include Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle Database, and SQL Server.

## Microsoft:

Azure SQL Database is a managed database service which is different from AWS RDS which is a container service. Microsoft Azure SQL Database includes built-in intelligence that learns app patterns and adapts to maximize performance, reliability, and data protection.

**Note: Everything online, No installations needed on the client machine.**

# Triggers

---

**Trigger is an action that the database takes automatically when an SQL DML event is performed. A trigger contains an event, an action and timing.**

- Trigger is an SQL code executed automatically in response to a certain event on table
- It is an associated action that the database will take automatically when a specified activity is performed on a table
- This trigger gets executed once data is affected by an insert/update/delete
- It is a kind of procedure that runs automatically after an activity
- It contains an event, an action and the timing
- When the event occurs the action is taken automatically before or after the event as specified

## **Conclusion:**

- Triggers maintain the integrity of transaction and associate activities so that nothing is missed out

```
create table acctrans
(
  accno int,
  trtype varchar(10),
  trdate datetime not null,
  amt float
);
```

DELIMITER //

```
create trigger transact
after insert
on acctrans
for each row
begin
if new.trtype='deposit' then
update accounts set balance=balance+new.amt where
accounts.accno=new.accno;
end if;
if new.trtype='withdraw' then
update accounts set balance=balance-new.amt where
accounts.accno=new.accno;
end if;
END//
```

DELIMITER ;

TEST IT USING:

```
insert into acctrans values(1003,'deposit',Now(),1200);
```

---

CREATE TABLE acchistory

```
(
  acno int,
  acnm varchar(30),
  actype varchar(20),
  balance float
);
```



DELIMITER //

```
create trigger copyacc  
after delete  
on accounts  
for each row  
begin  
insert into acchistory set  
acno=old.accno,acnm=old.accnm,actype=old.actype,balance=old.balance;  
END//
```

DELIMITER ;

```
delete from accounts where accno=1004;  
select * from acchistory;
```

---



**Sharayu, Training Head**

“ I agree, as a student, big complex tools and technologies are difficult to learn but once you go thru they offer less competition and bigger careers. My training department has always promoted tools that ensure future growth globally. Each tool you learn becomes a part of skill set and your profile for the future. Our training and certification programs have exactly the same in the last 20 years and our students are comfortably working in the best IT companies in the world.”

# Joins

---

**It is process of combining data of more than one tables to generate a single result set.  
Depending on the behaviour of the joins they are classified into different types.**

- It is process of combining data of more than one tables to generate a single result
- Data is joined on the basis of condition based on common values of columns

Types of joins:

- Inner join
- Left outer join
- Right outer join
- Full outer join
- Cross join
- Self join

create table products

```
(  
prodid varchar(5) primary key,  
prodnm varchar(50),  
company varchar(20),  
price float  
);
```

```
insert into products values('sp841','Superb','Skoda',2854600.00);  
insert into products values('ms488','Surface Pro','microsoft',129399.00);  
insert into products values('ip627','iphone 8','apple',64100.00);  
insert into products values('hp419','development laptop','HP',54600.00);  
insert into products values('ph618','washing machine','philips',23200.00);  
insert into products values('lt532','Smart FHD TV','samsung',36500.00);  
insert into products values('vn488','Vento','Volkswagen',1325000.00);  
insert into products values('x1562','X1 Suv','BMW',3469100.00);  
insert into products values('tg765','Tiguan','Volkswagen',2914900.00);  
insert into products values('gt261','6 Series GT','BMW',6323200.00);
```

```
select * from products;
```

```
create table customers(custno int primary key,custnm varchar(30),prodid varchar(5),paymode  
varchar(30) default 'credit card');
```

```
insert into customers(custno,custnm,prodid,paymode) values (11,'Joe  
Root','hp419','netbanking');
```

```
insert into customers(custno,custnm,prodid,paymode) values (22,'Boris Becker','tg765','cod');
```

```
insert into customers(custno,custnm,prodid) values (33,'Michael Schumacher','lt532');
```

```
insert into customers(custno,custnm,prodid) values (44,'Arnold Schwarzenegger','hp419');
```

```
insert into customers(custno,custnm,prodid) values (55,'Jos Buttler','lt532');
```

```
insert into customers(custno,custnm,prodid,paymode) values (66,'Rebecca  
Ferguson','ip627','netbanking');
```

```
insert into customers(custno,custnm,prodid,paymode) values (77,'Bill Gates','ms488','debit  
card');
```

```
insert into customers(custno,custnm,prodid) values (88,'Mohamed Salah','sg375');
```

```
insert into customers(custno,custnm,prodid) values (99,'Ferdinand Porsche','tg765');
```

```
insert into customers(custno,custnm,prodid,paymode) values (111,'Tom Cruise','ss210','cod');
```

```
insert into customers(custno,custnm,prodid) values (222,'Roger Federer','gt261');
```

```
insert into customers(custno,custnm,prodid) values (333,'Chris Woakes','dm298');
```

```
select * from customers;
```

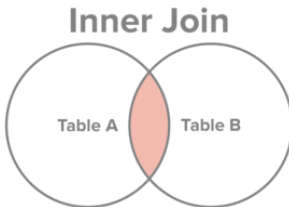
---

**Note: Create these two tables with the records using the provided insert queries to try all kinds of joins.**

## Inner Join

It returns only matching records from both the tables

- It delivers only the common matching records between the tables
- Returns rows that have matching values in both tables
- All mismatching records of both tables are discarded



```
select customers.custnm,products.prodnm,products.company,products.price
from customers
inner join
products on customers.prodid=products.prodid;
```

```
mysql> select customers.custnm,products.prodnm,products.company,products.price
-> from customers
-> inner join
-> products on customers.prodid=products.prodid;
```

custnm	prodnm	company	price
Joe Root	development laptop	HP	54600
Boris Becker	Tiguan	Volkswagen	2914900
Michael Schumacher	Smart FHD TV	samsung	36500
Arnold Schwarzenegger	development laptop	HP	54600
Jos Buttler	Smart FHD TV	samsung	36500
Rebecca Ferguson	iphone 8	apple	64100
Bill Gates	Surface Pro	microsoft	129399
Ferdinand Porsche	Tiguan	Volkswagen	2914900
Roger Federer	6 Series GT	BMW	6323200

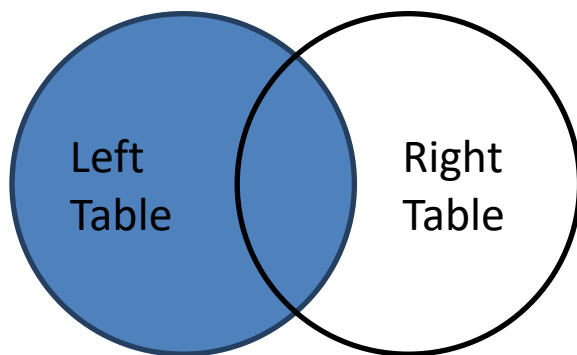
```
9 rows in set (0.07 sec)

mysql>
```

## Left outer join

Delivers all records from left side table with matching records on the right side, if no match is found, they are joined with a NULL record on the right side.

- It takes all records of the left side table with their matching records from the right side table
- If no match is found on the right, the records are joined with NULL
- It preserves the unmatched rows from the first (left) table, joining them with a NULL row in the shape of the second (right) table.



```
select customers.custnm,products.prodnm,products.company,products.price
from customers
left outer join
products on customers.prodid=products.prodid;
```

```
mysql> select customers.custnm,products.prodnm,products.company,products.price
-> from customers
-> left outer join
-> products on customers.prodid=products.prodid;
```

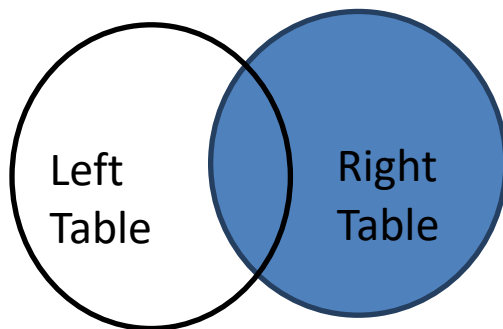
custnm	prodnm	company	price
Joe Root	development laptop	HP	54600
Boris Becker	Tiguan	Volkswagen	2914900
Michael Schumacher	Smart FHD TV	samsung	36500
Arnold Schwarzenegger	development laptop	HP	54600
Jos Buttler	Smart FHD TV	samsung	36500
Rebecca Ferguson	iphone 8	apple	64100
Bill Gates	Surface Pro	microsoft	129399
Mohamed Salah	NULL	NULL	NULL
Ferdinand Porsche	Tiguan	Volkswagen	2914900
Tom Cruise	NULL	NULL	NULL
Roger Federer	6 Series GT	BMW	6323200
Chris Woakes	NULL	NULL	NULL

```
12 rows in set (0.05 sec)
```

## Right outer Join

Delivers all records from right side table with matching records on the left side, if no match is found, they are joined with a NULL record on the left side.

- It takes all records of the Right side table with their matching records from the left side table
- If no match is found on the left, the records are joined with NULL
- right join returns all the values from the right table with matching values from the left table or NULL in case of no matching join



```
select customers.custnm,products.prodnm,products.company,products.price
from customers
right outer join
products on customers.prodid=products.prodid;
```

```
mysql> select customers.custnm,products.prodnm,products.company,products.price
-> from customers
-> right outer join
-> products on customers.prodid=products.prodid;
```

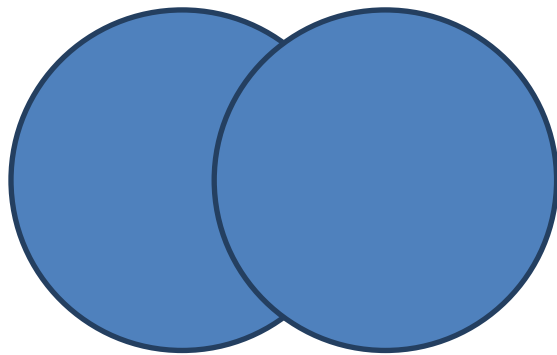
custnm	prodnm	company	price
Joe Root	development laptop	HP	54600
Boris Becker	Tiguan	Volkswagen	2914900
Michael Schumacher	Smart FHD TV	samsung	36500
Arnold Schwarzenegger	development laptop	HP	54600
Jos Buttler	Smart FHD TV	samsung	36500
Rebecca Ferguson	iphone 8	apple	64100
Bill Gates	Surface Pro	microsoft	129399
Ferdinand Porsche	Tiguan	Volkswagen	2914900
Roger Federer	6 Series GT	BMW	6323200
NULL	washing machine	philips	23200
NULL	Superb	Skoda	2854600
NULL	Vento	Volkswagen	1325000
NULL	X1 Suv	BMW	3469100

```
13 rows in set (0.01 sec)
```

## Full outer join (not supported by MySQL)

Delivers all records from both side tables with matching records, if no match is found, they are joined with a NULL record on either side.

- It takes all records from both left and right tables
- No record is left out, all values from both tables are included
- Rows are joined if they match else they are joined with NULL on the other side



```
select customers.custnm,products.prodnm,products.company,products.price
from customers
full outer join
products on customers.prodid=products.prodid;
```

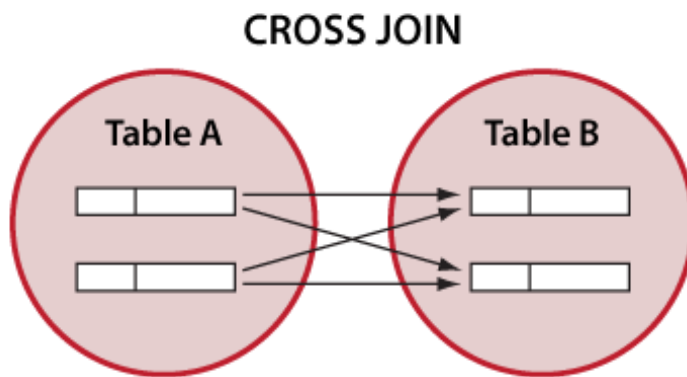
Query1.sql - DE...sharayudb (sa (59))			
<pre>select customers.custnm,products.prodnm,products.company,products.price from customers full outer join products on customers.prodid=products.prodid;</pre>			
Results			
custnm	prodnm	company	price
Amir Khan	development laptop	HP	54600
Boris Becker	Tiguan	Volkswagen	2914900
Michael Schumacher	Smart FHD TV	samsung	36500
Arnold Schwarzenegger	development laptop	HP	54600
Bear Grylls	Smart FHD TV	samsung	36500
Rebecca Ferguson	iphone 8	apple	64100
Bill Gates	Surface Pro	microsoft	129399
Mohamed Salah	NULL	NULL	NULL
Ferdinand Porsche	Tiguan	Volkswagen	2914900
Tom Cruise	NULL	NULL	NULL
Roger Federer	6 Series GT	BMW	6323200
Vladimir Putin	NULL	NULL	NULL
NULL	washing machine	philips	23200
NULL	Superb	Skoda	2854600
NULL	Vento	Volkswagen	1325000
NULL	X1 Suv	BMW	3469100

Output from  
MSSQL Server 2019

## Cross join

Every record in the left side table is joined with each record on the right side without considering any joining criteria.

- Each record in the left side table is joined with every record in the right side table
- A cross join is used when you wish to create a combination of every row from two tables.
- All row combinations are included in the result



```
select * from customers cross join products;
```

```
mysql> select customers.custnm,products.prodnm
-> from customers cross join products;
+-----+-----+
| custnm | prodnm |
+-----+-----+
| Amir Khan | 6 Series GT |
| Amir Khan | development laptop |
| Amir Khan | iphone 8 |
| Amir Khan | Smart FHD TV |
| Amir Khan | Surface Pro |
| Amir Khan | washing machine |
| Amir Khan | Superb |
| Amir Khan | Tiguan |
| Amir Khan | Vento |
| Amir Khan | X1 Suv |
| Rebecca Ferguson | 6 Series GT |
| Rebecca Ferguson | development laptop |
| Rebecca Ferguson | iphone 8 |
| Rebecca Ferguson | Smart FHD TV |
| Rebecca Ferguson | Surface Pro |
| Rebecca Ferguson | washing machine |
| Rebecca Ferguson | Superb |
| Rebecca Ferguson | Tiguan |
| Rebecca Ferguson | Vento |
| Rebecca Ferguson | X1 Suv |
+-----+-----+
```

Every customer with each product

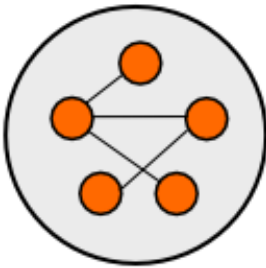


## Self join

### Data of a table is joined with itself

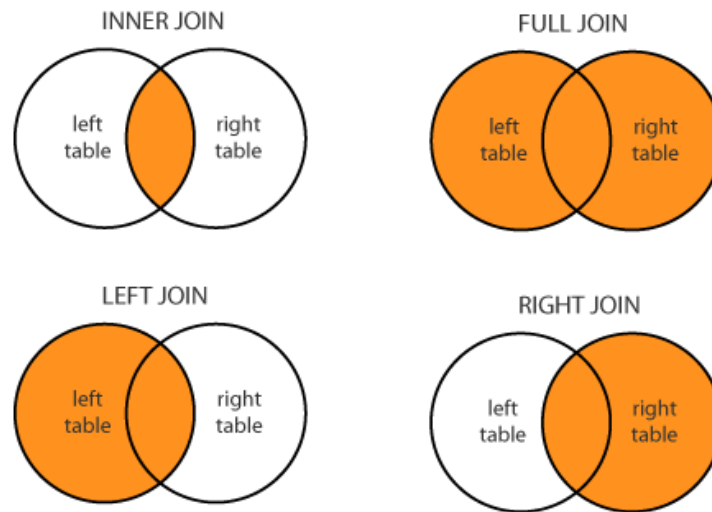
- Records of a table are joined with its own on the basis of a certain criteria
- Inner join is used in the syntax of the query
- There is no possibility of a mismatch, a record is joined with itself atleast

### Self Join



```
select c1.custnm,c2.custnm from customers c1
inner join customers c2
on c1.prodid=c2.prodid;
```

```
mysql> select c1.custnm,c2.custnm from customers c1
-> inner join customers c2
-> on c1.prodid=c2.prodid;
+-----+-----+
| custnm | custnm |
+-----+-----+
| Joe Root | Joe Root |
| Arnold Schwarzenegger | Joe Root |
| Boris Becker | Boris Becker |
| Ferdinand Porsche | Boris Becker |
| Michael Schumacher | Michael Schumacher |
| Jos Buttler | Michael Schumacher |
| Joe Root | Arnold Schwarzenegger |
| Arnold Schwarzenegger | Arnold Schwarzenegger |
| Michael Schumacher | Jos Buttler |
| Jos Buttler | Jos Buttler |
| Rebecca Ferguson | Rebecca Ferguson |
| Bill Gates | Bill Gates |
| Mohamed Salah | Mohamed Salah |
| Boris Becker | Ferdinand Porsche |
| Ferdinand Porsche | Ferdinand Porsche |
| Tom Cruise | Tom Cruise |
| Roger Federer | Roger Federer |
| Chris Woakes | Chris Woakes |
+-----+-----+
18 rows in set (0.00 sec)
```



### Conclusion:

- The information that the programs demand, is not every time present in a single table so joins provide a mechanism to combine data from more than one tables to generate the final result or report

## Index

---

- A database index is a data structure that improves the speed of data retrieval operations on a database table
- Indexes are used to quickly locate data without having to search every row in a database table every time a database table is accessed.
- It needs additional writes and storage space to maintain the index
- Indexes are related to specific tables and consist of one or more keys.
- Faster locating or searching of rows in the table improve performance of Select, Update, Delete and Joins
- It is essential the correct indexes are defined for each table based on the possible search criteria

### Types of Index:

- Non-clustered index: This type of index doesn't change the physical order of data in the table. There can be many logical non clustered indexes possible
- Clustered index: Data in the tables is physically arranged as per the order of the index.

INDEX on name

custnm
Amir
Praffull
Sharayu
Soham
Zaheer

TABLE

custno	custnm	mobile	e
1026	Soham	...	..
1153	Sharayu	...	..
1182	Zaheer	...	..
1270	Amir	...	..
1309	Praffull	...	..

Create index nmindex on accounts(accnm);



Select \* from accounts where accnm='soham';

Update accounts set balance=balance-150 where accnm='sharayu';

Delete from accounts where accnm='megha';

### Conclusion:

- We may not realize importance of having an index with smaller tables but with larger tables index is a must to improve performance of data operations

# Built-in Functions in SQL

---

SQL provides a large collection of ready to use functions to manipulate database data easily. These functions are of different categories.

- String Functions

Examples: `soundex()`, `Reverse()`, `Replace()`, `Upper()`, `Lower()`, `SubString()`, etc.

- Date/Time functions

Examples: `Curdate()`, `Curtime()`, `Datediff()`, `Adddate()`, `Addtime()`, etc.

- Numeric functions

Examples: `Sum()`, `Min()`, `Max()`, `Avg()`, `Count()`, `Round()`, `Sqrt()`, etc.

```
Select Curdate();
```

```
Select * from students where soundex(studnm)=soundex('prful');
```

```
Select adddate('2020-01-13',148);
```

**Watch video for detailed use of these function with a database table**

## Commit and Rollback: transactions

---

### Commit

- It is used to confirm a transaction to the database server.
- DDL queries don't need commit
- No need to manually commit if autocommit is ON.

### Rollback:

- It is used to cancel a transaction. An activity can't be rolled back if it is committed.
- A DDL activity can't be rolled back
- It cancels every activity till the last commit not just the last one. 2 inserts, 5 updates and 1 delete followed by a rollback will cancel all 8 activities not just one delete. Timely commit can help

**Watch video to understand proper use of commit and rollback.**

# Backup and Restore

---

## Backup

- Backup is a file that contains full copy of a database which includes everything like tables, data, procedures, etc.
- This backup can be used for safety purpose in case of an emergency or it can be used to transport the entire database to another server
- It enables the creation of a duplicate instance or copy of a database in case the primary database crashes, is corrupted or is lost.

`mysqldump -h localhost -u root -p databasename -R -E > backup_file_location\backupfilename.sql`

## Restore:

- It is process that restores data and all other objects from a backup file to another database server or on the same server with a different name

`mysql -h localhost -u root -p databasename < backup_file_location\backupfilename.sql`



Every database has a different way to take backup and restore it. Databases like Oracle and MSSQL server have graphical interfaces to do this.

## Conclusion:

- Maintaining regular database backups is always a very good idea. In case of any damage to the original database, the backup file comes to the rescue.





MICROSOFT PARTNER | ISO 9001:2015 CERTIFIED

# SOHAMGLOBAL

Training, Projects, Internships



## 20+ YEARS OF EXPERIENCE

Contact

**[sohamglobal.com](http://sohamglobal.com) | 9890925745**

3rd Floor, Shivaji Complex, Ravi Nagar Square  
AMRAVATI