Day 1

```
* Data, processing, Information
* Database
* Need Of Database
* Applications
* DBMS vs RDBMS
* MySql Introduction
```

Day 2

- Databases do not understand C/C++/Java language. It can understand only SQL("Sequel")[Structured Query Language].
- Initial name of SQL was RQBE(Relational Query By Example). It is introduced by IBM in 1975.
- ANSI is responsible for standardzing SQL. It means that SQL is common for all databases.
- In 2005, code of sql was rewritten in Java.
- Sub Division of SQL commands

1. Data Definition Language[DDL]
    - CREATE
    - ALTER
    - RENAME
    - DESCRIBE
    - DROP
    - TRUNCATE
2. Data Manipulation Language[DML]
    - INSERT
    - UPDATE
    - DELETE
3. Data Query Language[DQL]
    - SELECT
4. Transaction Control Language[TCL]
    - COMMIT
    - ROLLBACK
    - SAVEPOINT
5. Data Control Language[DCL]
    - GRANT
    - REVOKE

Naming Conventions for identifier[Database/ Table / Column name]

1. Maximum 30 characters are allowed
2. Name must begin with character

3. It can contain[A-Z,a-z,0-9,$,_]
4. Reservered word can not be used as identifier.
5. Identifiers are case insensitive.

```
mysql -u root -pmanager
```

```
SELECT user();
SELECT user() FROM DUAL;
```

- DUAL is single row and 2 column table.
- It is a dummy table name, we should use it situations where no tables are referenced:

```
SELECT 2 + 3;
SELECT 2 + 3 FROM DUAL;
```

- Comments

```
-- SELECT 5*5 FROM DUAL;
# SELECT 5*5 FROM DUAL;
/* SELECT 5*5 FROM DUAL; */
```

- CURRENT_USER() is a function which returns user name and host name combination for the MySQL account that the server used to authenticate the current client.
- Check exsting Users

```
SELECT User from user;
```

- Creating New User

```
CREATE USER
'dac'@'localhost'
IDENTIFIED BY
'dac';
```

- Delete User

```
DROP USER 'dac'@'localhost';
```

```
DELETE
FROM user
WHERE
User='dac';
```

- Check User Permissions

```
SHOW GRANTS FOR 'dac'@'localhost';
```

- Assigning permission's to user

```
-- GRANT ALL PRIVILEGES
GRANT ALL
ON
*.*
TO
'dac'@'localhost';
```

```
FLUSH PRIVILEGES;
```

- Removing permission's to user

```
-- REVOKE INSERT ON *.* FROM 'jeffrey'@'localhost';

-- REVOKE ALL PRIVILEGES
REVOKE ALL
ON
*.*
FROM
'dac'@'localhost';
```

```
FLUSH PRIVILEGES;
```

```
SHOW GRANTS FOR 'dac'@'localhost';
```

```
-- mysql -u dac -p
mysql -u dac -pdac
```

- In context of SQL, database is also called as schema.
- List databases:

```
SHOW DATABASES;
```

- Create new database

```
-- CREATE DATABASE dac_db;  -- or
CREATE SCHEMA dac_db;
```

- Check currenlty selected database

```
-- SELECT DATABASE() -- or
SELECT DATABASE() FROM DUAL;
```

```
-- SELECT SCHEMA() -- or
SELECT SCHEMA() FROM DUAL;
```

- Working with database

```
USE dac_db;
SELECT DATABASE() FROM DUAL;
```

- List tables from database

```
SHOW TABLES;
```

- Creating Table
- Syntax: CREATE TABLE tbl_name (col_name column_definition);

```
CREATE TABLE books
(
    id INT,
    name VARCHAR(50),
    author VARCHAR(50),
    subject VARCHAR(50),
    price FLOAT
);
```

```
SHOW TABLES;
```

- CHECK TABLE STRUCTURE

```
DESCRIBE books; -- or
DESC books; -- or
EXPLAIN books;  -- or
SHOW COLUMNS FROM books;
```

- Rename Table

```
RENAME TABLE books TO books_tbl;
```

- If we want to modify table structure then we should use ALTER statement.
- How to rename column?
    - Syntax: ALTER TABLE tbl_name RENAME COLUMN old_col_name TO new_col_name;

```
ALTER TABLE books
RENAME COLUMN id TO book_id;

ALTER TABLE books
RENAME COLUMN name TO book_name;

ALTER TABLE books
RENAME COLUMN author TO author_name;

ALTER TABLE books
RENAME COLUMN subject TO subject_name;
```

- How to modify column Definition?
    - Syntax ALTER TABLE tbl_name MODIFY [COLUMN] col_name column_definition.

```
ALTER TABLE books
MODIFY COLUMN book_name VARCHAR(256 );

ALTER TABLE books
MODIFY author_name VARCHAR(256 );

ALTER TABLE books
MODIFY price DOUBLE;

ALTER TABLE books
MODIFY book_id INT(5);
```

- How to add new column in table?
    - Syntax: ALTER TABLE tbl_name ADD [COLUMN] col_name column_definition

```
ALTER TABLE books
ADD COLUMN pub_name VARCHAR(50);
```

- How to drop column from table?
    - Syntax: ALTER TABLE tbl_name DROP [COLUMN] col_name

```
ALTER TABLE books
DROP COLUMN pub_name;
```

- How to insert record into table?
    - Syntax: INSERT INTO tbl_name (col_name1 , col_name2 ...) VALUES (value_list)

```
INSERT INTO books
( book_id, book_name, author_name, subject_name, price )
VALUES
( 1, 'Let Us C', 'Yashwant Kanetkar', 'C', 450 );
```

```
INSERT INTO books
VALUES
( 2, "More Effective C++", "Scott Mayers", "C++", 550 );
```

```
INSERT INTO books
(book_id, subject_name, book_name, author_name, price )
VALUES
( 3, 'Java','Java Certification', 'Khalid Mughal', 650 );
```

```
INSERT INTO books
(book_id, book_name , price )
VALUES
( 4, 'CLR Via C#', 850 );
```

```
INSERT INTO books
VALUES
( 5, 'OS Concepts',null,NULL,500 );
```

```
INSERT INTO books()VALUES( );
```

```
INSERT INTO books
VALUES
(6,'The C Prog Lang.','Dennis Ritche','C', 450),
(7,'C++ Complete Reference','Herbert Schildt', 'C++', 600),
(8,'Java Head First', 'Kathy Siera','Java', 800);
```

- How to view records/rows?
  - SELECT is used to retrieve rows selected from one or more tables
  - Syntax SELECT FROM table_references;

```
SELECT * FROM books;
```

- How to create copy of table?

```
CREATE TABLE new_books
AS
SELECT * FROM books;
```

```
CREATE TABLE new_books_tbl
AS
SELECT book_id, subject_name, book_name, author_name, price FROM books;
```

- How to copy table structure?

```
CREATE TABLE book_table LIKE books;
```

```
INSERT INTO book_table
( SELECT * FROM books );
```

- How to import sql file?

```
CREATE DATABASE classwork;
USE classwork;
```

```
--SOURCE (Drag and Drop ) .sql file here
SOURCE /Users/sandeepkulange/Desktop/DBT/classwork-db.sql;
```

- Fetch records from all columns

```
SELECT
id, name, author, subject, price
FROM
books;
```

```
SELECT
*
FROM
books;
```

- Fetch records from few columns

```
SELECT
name, author, price
FROM
books;
```

```
SELECT
-- name, author, price, price + price * 0.10
-- name, author, price, price + price * 0.10 AS Final_Price
-- name, author, price, price + price * 0.10 AS "Final Price"
-- name, author, price, price + price * 0.10 AS 'Final Price'
-- name, author, price, price + price * 0.10 'Final Price'
name Name, author Author, price Price, price + price * 0.10 'Final Price'
FROM
books;
```

- DISTINCT

```
-- SELECT  author  FROM books;
SELECT DISTINCT author  FROM books;
```

- LIMIT

```
-- SELECT * FROM books LIMIT 4;
SELECT * FROM books LIMIT 4,3;
```

```
SELECT
*
FROM books
-- ORDER BY price ASC;
-- ORDER BY price;
ORDER BY price DESC;
```

```
SELECT
*
FROM books
-- ORDER BY subject, author;
-- ORDER BY subject ASC, author DESC;
ORDER BY subject DESC, author ASC;
```

```
SELECT
*
FROM books
-- ORDER BY subject;
ORDER BY 4;
```

```
SELECT
id, name, author, price, price + price * 0.10 Final
FROM books
-- ORDER BY Final;
-- ORDER BY Final DESC;
ORDER BY 5 DESC;
```

- How to delete table?
    - DROP TABLE removes one or more tables.
    - Syntax: DROP TABLE tbl_name;
    - It is a DDL statement, which removes table data as well as table structure.

```
DROP TABLE new_books_tbl;
DROP TABLE book_table, new_books;
```

- How to truncate records?
    - TRUNCATE TABLE empties a table completely.
    - Logically, TRUNCATE TABLE is similar to a DELETE statement that deletes all rows, or a sequence of DROP TABLE and CREATE TABLE statements.

- Syntax : TRUNCATE TABLE tbl_name;

```
TRUNCATE TABLE books;
```

Day 3

Where clause

```
mysql -u dac -p
mysql -u dac -pdac
mysql -u dac -pdac classwork
```

- System Information Functions

1. DATABASE()

```
SELECT DATABASE() FROM DUAL;
```

2. USER

```
SELECT USER() FROM DUAL;
```

3. VERSION

```
SELECT VERSION() FROM DUAL;
```

```
SELECT DATABASE(),VERSION(),USER() FROM DUAL;
```

- If we want filter rows then we should use where clause.
- Operators
  - Arithmetic Operators : ( ), / ,* ,+, -
  - Relational Operators : <, <=, >, >=, ==, != / <>
  - BETWEEN, IN, LIKE
  - Logical Operators : AND, OR, NOT
- Get record of books from "books" table whose subject is "Java Programming"

```

```

```
SELECT * FROM books
WHERE subject='Java Programming';
```

```
SELECT * FROM books
WHERE subject="Java Programming";
```

```
SELECT * FROM books
WHERE subject="java programming";
```

- Find book(s) of 'Dennis Ritchie' from 'books' table;

```
SELECT * FROM books
WHERE author='Dennis Ritchie';
```

```
SELECT * FROM books
WHERE id=3001;
```

- Find book(s) of 'Herbert Schildt' Whose price is greater than 500.

```
SELECT * FROM books;

SELECT * FROM books
WHERE author='Herbert Schildt';

SELECT * FROM books
WHERE author='Herbert Schildt' AND price > 500;
```

- List book(s) of 'Operating Systems' or list book(s) whose author s 'Herbert Schildt' and price is greater than 500.

```
SELECT * from books;

SELECT * from books
WHERE subject='Operating Systems';

SELECT * FROM books
WHERE author='Herbert Schildt' AND price > 500;

SELECT * FROM books
```

```
WHERE subject='Operating Systems' OR author='Herbert Schildt' AND price >
500;

SELECT * FROM books
WHERE subject='Operating Systems' OR ( author='Herbert Schildt' AND price
> 500);

SELECT * FROM books
WHERE NOT subject='Operating Systems' OR author='Herbert Schildt' AND
price > 500;

SELECT * FROM books
WHERE  ( subject='Operating Systems' OR author='Herbert Schildt' )AND
price > 500;

SELECT * FROM books
WHERE  NOT ( subject='Operating Systems' OR author='Herbert Schildt' )AND
price > 500;
```

- Find employee(s) from EMP table whose manager is NULL

```
SELECT * FROM EMP;

SELECT * FROM EMP
WHERE mgr IS NULL;
```

- Find employee(s) from EMP table whose comm is not NULL

```
SELECT * FROM emp
WHERE comm IS NOT NULL;
```

- Find employee(s) from EMP table whose sal is in between 2000 and 3000.

```
SELECT * FROM emp
-- WHERE sal >= 2000 AND sal <= 3000;
WHERE sal BETWEEN 2000 AND 3000;
```

- Find employee(s) from EMP table whose sal is in not between 2000 and 3000.

```
SELECT * FROM emp
WHERE sal NOT BETWEEN 2000 AND 3000;
```

```
SELECT * FROM emp
WHERE ename BETWEEN 'D' AND 'M';
```

- We can use BETWEEN operator to compare number, date and string.

```
SELECT * FROM emp;

SELECT * FROM emp
WHERE deptno=10;

SELECT * FROM emp
WHERE deptno=20;

SELECT * FROM emp
WHERE deptno=30;

SELECT * FROM emp
WHERE deptno=40;
```

- List all the employee(s) whose dept no is either 10, 20 or 30

```
SELECT * FROM emp
WHERE deptno=10 OR deptno=20 OR deptno=30;
```

```
SELECT * FROM emp
-- WHERE deptno IN (10, 20, 30 );
WHERE deptno NOT IN (10, 20, 30 );
```

- Find Details of Employee which is having third highest salary.

```
SELECT * FROM emp;

SELECT * FROM emp
ORDER BY sal DESC;

SELECT * FROM emp
ORDER BY sal DESC
LIMIT 2,1;
```

- If want to search values in column using specific pattern then we should use LIKE operator.
- ▢)
- Print Details of Java Books whose name starts with 'Java'.

```
SELECT * FROM books;

SELECT * FROM books
WHERE name LIKE 'Java%';
```

- Print Details of Java Books whose name ends with 'Language'.

```
SELECT * FROM books;

SELECT * FROM books
WHERE name LIKE '%Language';
```

Print Details of Java Books whose name contains 'Programming'.

```
SELECT * FROM books;

SELECT * FROM books
WHERE name LIKE '%Programming%';
```

```
SELECT * FROM books;

SELECT * FROM books
WHERE name LIKE '%in%';
```

```
SELECT * FROM books;

SELECT * FROM books
WHERE name LIKE 'J%e';
```

```
SELECT * FROM books
WHERE name LIKE '_N%';
```

```
SELECT * FROM books
WHERE name LIKE '%e_';
```

Update Statement

- If we want to update column definition then we should use ALTER statement.

- If we want to update record/row then we should use UPDATE statement.
- Syntax: UPDATE table_reference SET assignment_list;
- Update book name from books table whose book id is 1026.

```
UPDATE books
SET book_name = 'Linux Programming Interface'
WHERE book_id = 1026;
```

- Update subject name and author name from books table whose book id is 1026.

```
UPDATE books
SET subject_name='OS', author_name='Michael Kerrisk'
WHERE book_id=1026;
```

- Update price of C++ books by 5%.

```
UPDATE books
SET price=price - price * 0.05
WHERE subject_name='C++';
```

- Update price of all books by 5%.

```
UPDATE books
SET price = price - price * 0.05;
```

Delete Statement

- DELETE is a DML statement that removes rows from a table.
- Syntax : DELETE FROM tbl_name;
- Delete book from table whose book id is 1016.

```
DELETE FROM books
WHERE book_id = 1016;
```

- Delete all the books from table whose subject name is 'AWP'

```
DELETE FROM books
WHERE subject_name='AWP';
```

- Delete all the books from table

```
DELETE FROM books;
```

Function

- If we want to process data stored in row then we can use function.
- Types:
    1. Single Row Function
    2. Multi Row Function / Group Function / Aggregate Function
- Functions
    - Single Row Functions
        - String Functions (URL: http://dev.mysql.com/doc/refman/8.0/en/string-functions.html) 1 length : LENGTH(str) 2 concat : CONCAT(str1,str2,...) 3 upper : UPPER(str) 4 lower : LOWER(str) 5 left : LEFT(str,len) 6 right : RIGHT(str,len) 7 substr SUBSTRING(str,pos), SUBSTRING(str FROM pos), SUBSTRING(str,pos,len), SUBSTRING(str FROM pos FOR len) 8 lpad : LPAD(str,len,padstr) 9 rpad : RPAD(str,len,padstr) 10 ltrim : LTRIM(str) 11 rtrim : RTRIM(str) 12 trim : TRIM([{BOTH | LEADING | TRAILING} [remstr] FROM] str), TRIM([remstrFROM] str) 13 replace : REPLACE(str, FROM_str, TO_STR) 14 reverse : REVERSE(str) 15 instr : INSTR(str,substr) 16 ascii : ASCII(str) 17 char 18 bin : BIN(N)
        - Numeric Functions(URL: http://dev.mysql.com/doc/refman/8.0/en/mathematical-functions.html)
            1. Round : ROUND(X), ROUND(X,D)
            2. Truncate : TRUNCATE(X,D)
            3. Floor : FLOOR(X)
            4. Ceil : CEILING(X)
            5. Sign : SIGN(X)
            6. Mod : MOD(N,M), N % M, N MOD M
            7. Sqrt : SQRT(X)
            8. Power : POW(X,Y)
            9. Abs : ABS(X)
            10. Log : LOG(X), LOG(B,X)
            11. Sin : SIN(X), cos, tan
        - Date Functions(URL: http://dev.mysql.com/doc/refman/8.0/en/date-and-time-functions.html)
            1. sysdate : SYSDATE([fsp])
            2. Now : NOW([fsp])
            3. Sleep : SLEEP(duration)
            4. Adddate : ADDDATE(date,INTERVAL expr unit), ADDDATE(expr,days)
            5. Date_add: DATE_ADD(date,INTERVAL expr unit), DATE_SUB(date,INTERVAL expr unit)

6. Datediff : DATEDIFF(expr1,expr2)
7. Lastday
8. Dayname
9. Monthname
10. Day, month, year
11. add-time
12. Date()
13. Time()
14. date_format

- List Functions(URL: http://dev.mysql.com/doc/refman/8.0/en/comparison-operators.html)
  1. Greatest
  2. Least
  3. Isnull
  4. Strcmp
- Control flow functions(https://dev.mysql.com/doc/refman/8.0/en/control-flow-functions.html)
  1. if()
  2. Ifnull
  3. nullif()

- Multi Row Functions
  - Group / Aggregate functions(https://dev.mysql.com/doc/refman/8.0/en/group-by-functions.html) 1.sum 2.Avg 3.min 4.max 5.count

1. Can not use nonaggregated/ regular column with group function.
2. Can not use single row function with group function.
3. Can not use group function in where clause.
4. Can not nest group function.