INDEX SHEET

Aim:

Basic ToDos :

● Design the logical view using ER Diagrams with tools

● Design Enhanced ER diagram using Workbench

● Forward Engineer your EER diagrams n Workbench

● Design Test Cases/ SQL Queries to demonstrate the working (or) Implement a GUI (Python/Java/Web) to demonstrate the database.

Blogging Platform:

a. Set up a database for blog posts, categories, tags, and user comments.

b. Implement SQL queries to display popular blog posts, manage comments, and categorize posts.

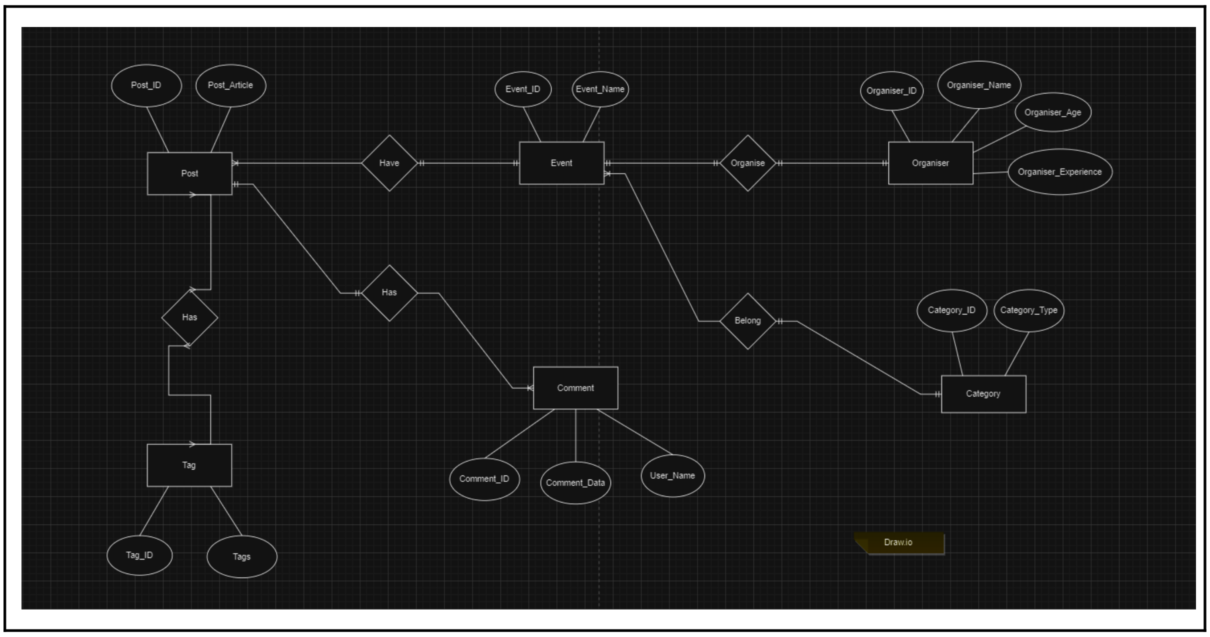
List of Entities & Attributes:

Entities	**Attributes**
Event	Event_ID, Event_Name
Post	Post_ID, Post_Article
Tag	Tag_ID, Tags
Category	Category_ID, Category_Type
Comment	Comment_ID, Comment_Data, User_Name
Organiser	Organiser_ID, Organiser_Name, Organiser_Age, Organiser_Experience

List of Relationships:

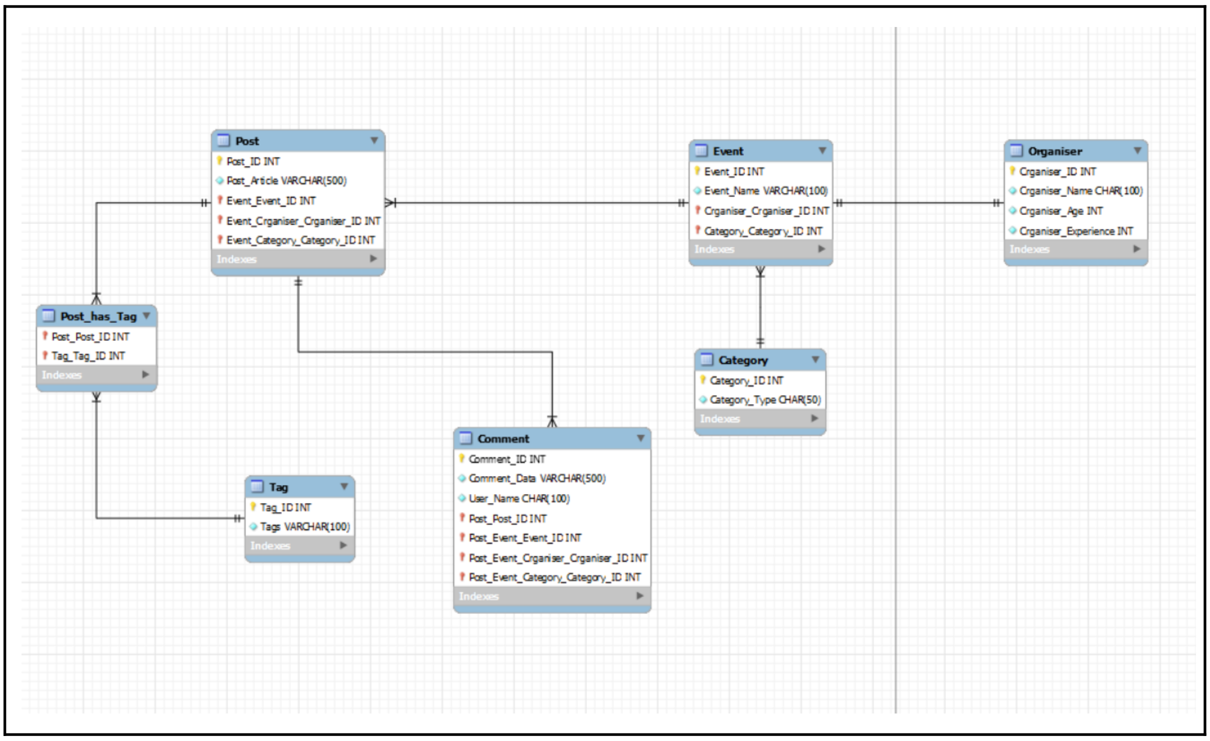
- Only one Organiser can organise only one Event.
- Many Events can belongs to one Category.
- One Event can have many Posts.
- One or many Posts can have one or many Tags.
- One Post can have many Comments.

ER Diagram:



Post_ID
Post_Article
Event_ID
Event_Name
Organiser_ID
Organiser_Name
Organiser_Age
Organiser_Experience
Post
Have
Event
Organise
Organiser
Has
Has
Belong
Category_ID
Category_Type
Category
Comment
Tag
Comment_ID
Comment_Data
User_Name
Tag_ID
Tags
Draw.io

EER Diagram:


Post
Post_ID INT
Post_Article VARCHAR(500)
Event_Event_ID INT
Event_Organiser_Organiser_ID INT
Event_Category_Category_ID INT
Indexes
Event
Event_ID INT
Event_Name VARCHAR(100)
Organiser_Organiser_ID INT
Category_Category_ID INT
Indexes
Organiser
Organiser_ID INT
Organiser_Name CHAR(100)
Organiser_Age INT
Organiser_Experience INT
Indexes
Post_has_Tag
Post_Post_ID INT
Tag_Tag_ID INT
Indexes
Category
Category_ID INT
Category_Type CHAR(50)
Indexes
Comment
Comment_ID INT
Comment_Data VARCHAR(500)
User_Name CHAR(100)
Post_Post_ID INT
Post_Event_Event_ID INT
Post_Event_Organiser_Organiser_ID INT
Post_Event_Category_Category_ID INT
Indexes
Tag
Tag_ID INT
Tags VARCHAR(100)
Indexes

Schema:

```
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_D
ATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTIT
UTION';

-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------


-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;
USE `mydb` ;

-- -----------------------------------------------------
-- Table `mydb`.`Organiser`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Organiser` (
  `Organiser_ID` INT NOT NULL,
  `Organiser_Name` CHAR(100) NOT NULL,
  `Organiser_Age` INT NOT NULL,
  `Organiser_Experience` INT NOT NULL,
  PRIMARY KEY (`Organiser_ID`),
  UNIQUE INDEX `Organiser_ID_UNIQUE` (`Organiser_ID` ASC) VISIBLE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`Category`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Category` (
  `Category_ID` INT NOT NULL,
  `Category_Type` CHAR(50) NOT NULL,
  PRIMARY KEY (`Category_ID`),
  UNIQUE INDEX `Category_ID_UNIQUE` (`Category_ID` ASC) VISIBLE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`Event`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Event` (
  `Event_ID` INT NOT NULL,
  `Event_Name` VARCHAR(100) NOT NULL,
  `Organiser_Organiser_ID` INT NOT NULL,
  `Category_Category_ID` INT NOT NULL,
  PRIMARY KEY (`Event_ID`, `Organiser_Organiser_ID`, `Category_Category_ID`),
  UNIQUE INDEX `Event_ID_UNIQUE` (`Event_ID` ASC) VISIBLE,
  INDEX `fk_Event_Organiser_idx` (`Organiser_Organiser_ID` ASC) VISIBLE,
  INDEX `fk_Event_Category1_idx` (`Category_Category_ID` ASC) VISIBLE,
  CONSTRAINT `fk_Event_Organiser`
    FOREIGN KEY (`Organiser_Organiser_ID`)
    REFERENCES `mydb`.`Organiser` (`Organiser_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Event_Category1`
    FOREIGN KEY (`Category_Category_ID`)
    REFERENCES `mydb`.`Category` (`Category_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`Post`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Post` (
  `Post_ID` INT NOT NULL,
  `Post_Article` VARCHAR(500) NOT NULL,
  `Event_Event_ID` INT NOT NULL,
  `Event_Organiser_Organiser_ID` INT NOT NULL,
  `Event_Category_Category_ID` INT NOT NULL,
  PRIMARY KEY (`Post_ID`, `Event_Event_ID`, `Event_Organiser_Organiser_ID`,
`Event_Category_Category_ID`),
  UNIQUE INDEX `Post_ID_UNIQUE` (`Post_ID` ASC) VISIBLE,
  INDEX `fk_Post_Event1_idx` (`Event_Event_ID` ASC,
`Event_Organiser_Organiser_ID` ASC, `Event_Category_Category_ID` ASC) VISIBLE,
  CONSTRAINT `fk_Post_Event1`
    FOREIGN KEY (`Event_Event_ID` , `Event_Organiser_Organiser_ID` ,
`Event_Category_Category_ID`)
    REFERENCES `mydb`.`Event` (`Event_ID` , `Organiser_Organiser_ID` ,
`Category_Category_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`Tag`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Tag` (
  `Tag_ID` INT NOT NULL,
  `Tags` VARCHAR(100) NOT NULL,
  UNIQUE INDEX `Tag_ID_UNIQUE` (`Tag_ID` ASC) VISIBLE,
  PRIMARY KEY (`Tag_ID`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`Comment`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Comment` (
  `Comment_ID` INT NOT NULL,
  `Comment_Data` VARCHAR(500) NOT NULL,
  `User_Name` CHAR(100) NOT NULL,
  `Post_Post_ID` INT NOT NULL,
  `Post_Event_Event_ID` INT NOT NULL,
  `Post_Event_Organiser_Organiser_ID` INT NOT NULL,
  `Post_Event_Category_Category_ID` INT NOT NULL,
  PRIMARY KEY (`Comment_ID`, `Post_Post_ID`, `Post_Event_Event_ID`,
`Post_Event_Organiser_Organiser_ID`, `Post_Event_Category_Category_ID`),
  UNIQUE INDEX `Comment_ID_UNIQUE` (`Comment_ID` ASC) VISIBLE,
  INDEX `fk_Comment_Post1_idx` (`Post_Post_ID` ASC, `Post_Event_Event_ID` ASC,
`Post_Event_Organiser_Organiser_ID` ASC, `Post_Event_Category_Category_ID` ASC)
VISIBLE,
  CONSTRAINT `fk_Comment_Post1`
    FOREIGN KEY (`Post_Post_ID` , `Post_Event_Event_ID` ,
`Post_Event_Organiser_Organiser_ID` , `Post_Event_Category_Category_ID`)
    REFERENCES `mydb`.`Post` (`Post_ID` , `Event_Event_ID` ,
`Event_Organiser_Organiser_ID` , `Event_Category_Category_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`Post_has_Tag`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Post_has_Tag` (
  `Post_Post_ID` INT NOT NULL,
  `Tag_Tag_ID` INT NOT NULL,
  PRIMARY KEY (`Post_Post_ID`, `Tag_Tag_ID`),
  INDEX `fk_Post_has_Tag_Tag1_idx` (`Tag_Tag_ID` ASC) VISIBLE,
  INDEX `fk_Post_has_Tag_Post1_idx` (`Post_Post_ID` ASC) VISIBLE,
  CONSTRAINT `fk_Post_has_Tag_Post1`
    FOREIGN KEY (`Post_Post_ID`)
    REFERENCES `mydb`.`Post` (`Post_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Post_has_Tag_Tag1`
    FOREIGN KEY (`Tag_Tag_ID`)
    REFERENCES `mydb`.`Tag` (`Tag_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```