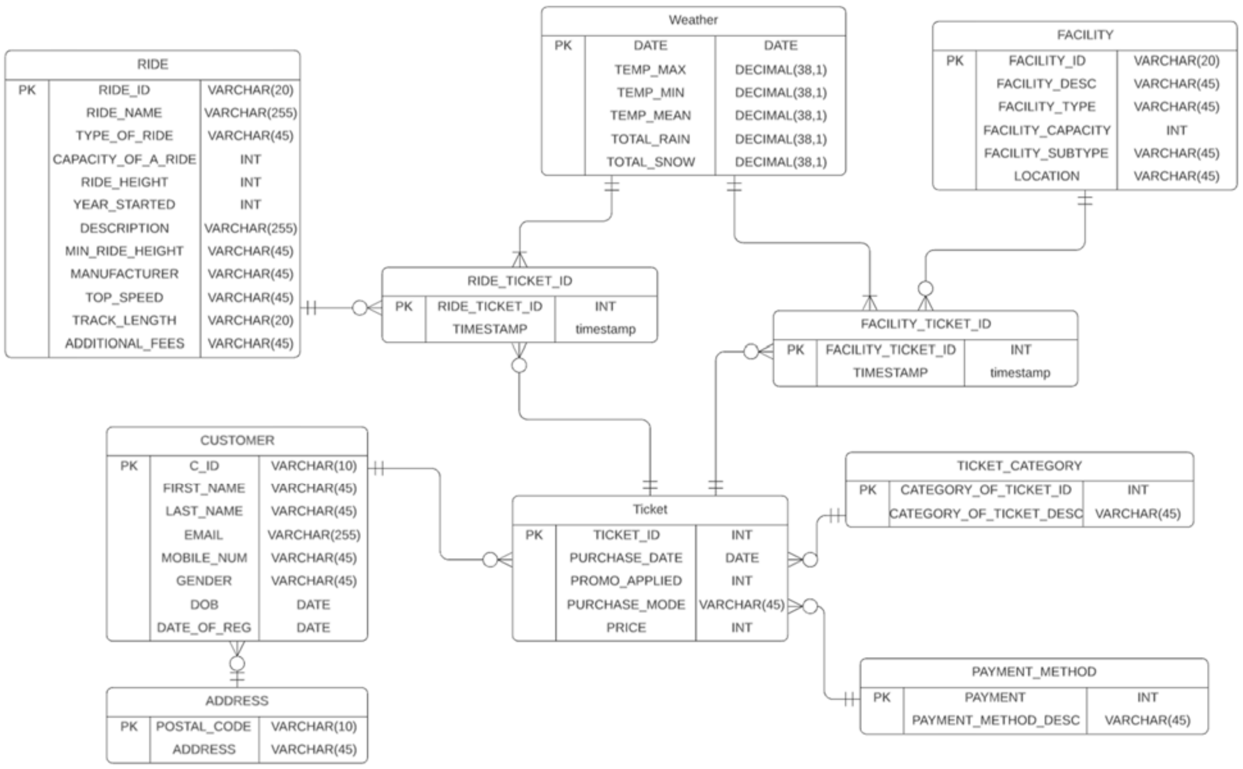
INSY 661 INDIVIDUAL PROJECTS (EXTERNAL DATA REPORT)

Yichen Wang

260761601

For external data, the weather data cited from the website of Government of Canada is used (https://climate.weather.gc.ca/climate_data) and all the daily weather data for 2020 is downloaded. The following extra tables and queries will be based on this.

1) **(20%) Revised ERD**:



The newly added table Weather contains the date as well as different weather attributes. By connecting it to the RIDE_TICKET_ID and FACILITY_TICKET_ID, we can find out the relationship between the weather and the visiting history of customers to different rides and facilities.

2) **(10%) RELATIONAL MODEL**: You should then create a relational model based on your ERD.

Use DDL to create tables in a MySQL database, as per your relational model. Link related tables together with foreign key constraints.

Revised:

RIDE (RIDE_ID, RIDE_NAME, TYPE_OF_RIDE, CAPACITY_OF_A_RIDE, RIDE_HEIGHT, YEAR_STARTED, DESCRIPTION, MIN_RIDE_HEIGHT, MANUFACTURER, TOP_SPEED, TRACK_LENGTH, ADDITIONAL_FEES)

FACILITY (FACILITY_ID, FACILITY_DESC, FACILITY_TYPE, FACILITY_CAPACITY, FACILITY_SUBTYPE, LOCATION)

TICKET_CATEGORY (CATEGORY_OF_TICKET_ID, CATEGORY_OF_TICKET_DESC)

PAYMENT_METHOD (PAYMENT, PAYMENT_METHOD_DESC)

ADDRESS (POSTAL_CODE, ADDRESS)

Weather (DATE, TEMP_MAX, TEMP_MIN, TEMP_MEAN, TOTAL_RAIN, TOTAL_SNOW**)**

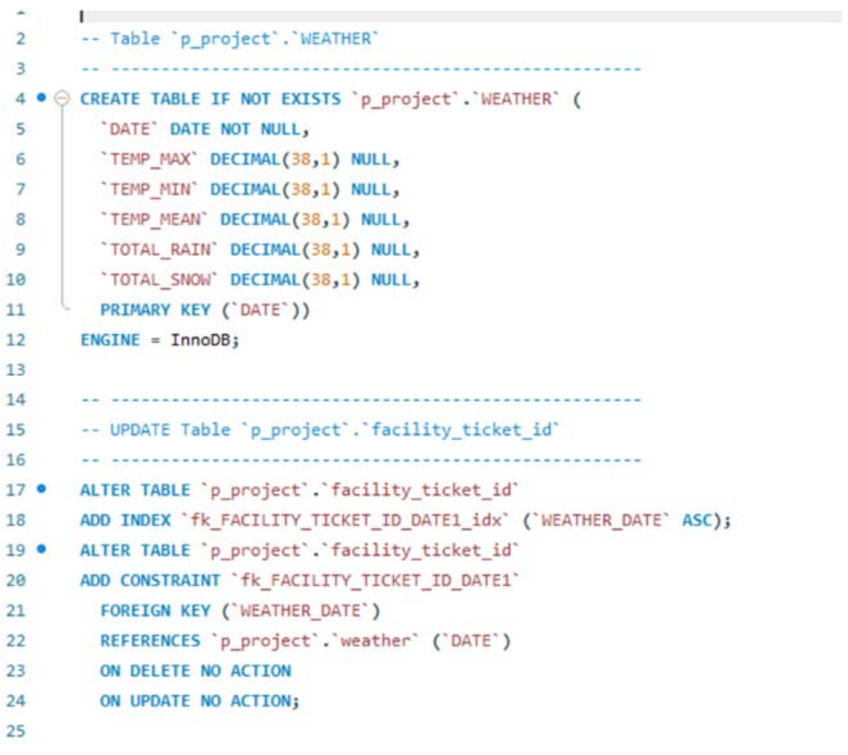
CUSTOMER (C_ID, FIRST_NAME, LAST_NAME, EMAIL, MOBILE_NUM, GENDER, DOB, DATE_OF_REG, POSTAL_CODE)

TICKET (TICKET_ID, PURCHASE_DATE, PROMO_APPLIED, PURCHASE_MODE, PRICE, CATEGORY_OF_TICKET_ID, PAYMENT, C_ID)

FACILITY_TICKET_ID (FACILITY_TICKET_ID, TIMESTAMP, TICKET_ID, FACILITY_ID, DATE)

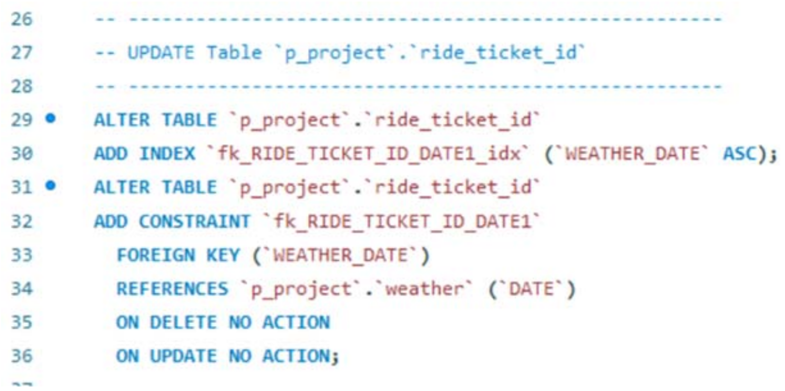
RIDE_TICKET_ID (RIDE_TICKET_ID, TIMESTAMP, TICKET_ID, RIDE_ID, DATE)

3) **(10%) POPULATE TABLES**: Use the data provided in the given CSV files to populate the tables that you have created in your database. NOTE: You should import all the files as tables in a MySQL database. These will act as temporary tables (youo can delete them later) that will be used to feed data (using INSERT) into tables that you have created as per your relational schema.



```
-- Table `p_project`.`WEATHER`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `p_project`.`WEATHER` (
  `DATE` DATE NOT NULL,
  `TEMP_MAX` DECIMAL(38,1) NULL,
  `TEMP_MIN` DECIMAL(38,1) NULL,
  `TEMP_MEAN` DECIMAL(38,1) NULL,
  `TOTAL_RAIN` DECIMAL(38,1) NULL,
  `TOTAL_SNOW` DECIMAL(38,1) NULL,
  PRIMARY KEY (`DATE`))
ENGINE = InnoDB;

-- -----------------------------------------------------
-- UPDATE Table `p_project`.`facility_ticket_id`
-- -----------------------------------------------------
ALTER TABLE `p_project`.`facility_ticket_id`
ADD INDEX `fk_FACILITY_TICKET_ID_DATE1_idx` (`WEATHER_DATE` ASC);
ALTER TABLE `p_project`.`facility_ticket_id`
ADD CONSTRAINT `fk_FACILITY_TICKET_ID_DATE1`
  FOREIGN KEY (`WEATHER_DATE`)
  REFERENCES `p_project`.`weather` (`DATE`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION;
```



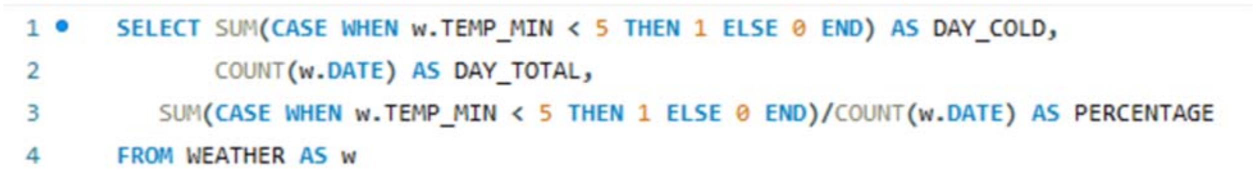
```
-- -----------------------------------------------------
-- UPDATE Table `p_project`.`ride_ticket_id`
-- -----------------------------------------------------
ALTER TABLE `p_project`.`ride_ticket_id`
ADD INDEX `fk_RIDE_TICKET_ID_DATE1_idx` (`WEATHER_DATE` ASC);
ALTER TABLE `p_project`.`ride_ticket_id`
ADD CONSTRAINT `fk_RIDE_TICKET_ID_DATE1`
  FOREIGN KEY (`WEATHER_DATE`)
  REFERENCES `p_project`.`weather` (`DATE`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION;
```

Since all the data has been cleaned using python before importing into the database, all the data insertion process will not be shown as only Import Wizard of the MySQL workbench is used.

4) **(20%) Queries: Prepare 10 SQL queries (Complex) that will give interesting insights** (e.g., derived value, aggregate functions, etc.).

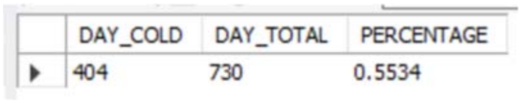
Q1. Customers usually won't come to the amusement park when the lowest temperature is below 5 degree Celsius. Find out the proportion of the days where customers don't want to come because of the weather in the past 2 years.

Codes:



```sql
SELECT SUM(CASE WHEN w.TEMP_MIN < 5 THEN 1 ELSE 0 END) AS DAY_COLD,
       COUNT(w.DATE) AS DAY_TOTAL,
    SUM(CASE WHEN w.TEMP_MIN < 5 THEN 1 ELSE 0 END)/COUNT(w.DATE) AS PERCENTAGE
FROM WEATHER AS w
```

Results:

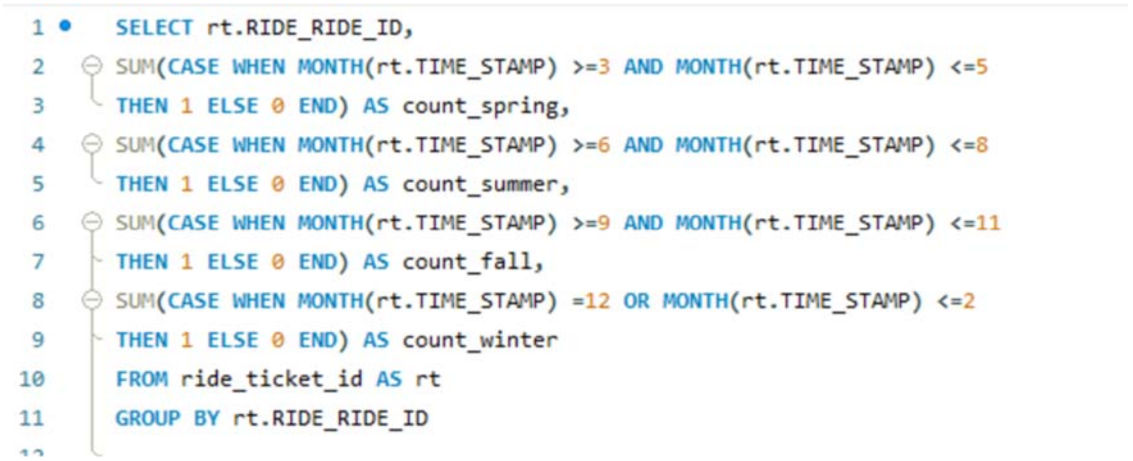


	DAY_COLD	DAY_TOTAL	PERCENTAGE
▶	404	730	0.5534

Q2. Find the number of visits for each rides in spring, summer, fall and winter

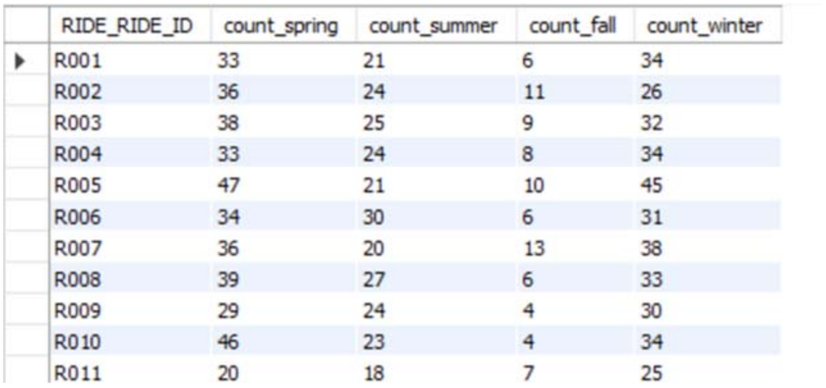
Assumption: spring is from march to may, summer is from june to august, fall is from september to november, winter is from december to feburary.

Codes:



```sql
SELECT rt.RIDE_RIDE_ID,
SUM(CASE WHEN MONTH(rt.TIME_STAMP) >=3 AND MONTH(rt.TIME_STAMP) <=5
THEN 1 ELSE 0 END) AS count_spring,
SUM(CASE WHEN MONTH(rt.TIME_STAMP) >=6 AND MONTH(rt.TIME_STAMP) <=8
THEN 1 ELSE 0 END) AS count_summer,
SUM(CASE WHEN MONTH(rt.TIME_STAMP) >=9 AND MONTH(rt.TIME_STAMP) <=11
THEN 1 ELSE 0 END) AS count_fall,
SUM(CASE WHEN MONTH(rt.TIME_STAMP) =12 OR MONTH(rt.TIME_STAMP) <=2
THEN 1 ELSE 0 END) AS count_winter
FROM ride_ticket_id AS rt
GROUP BY rt.RIDE_RIDE_ID
```

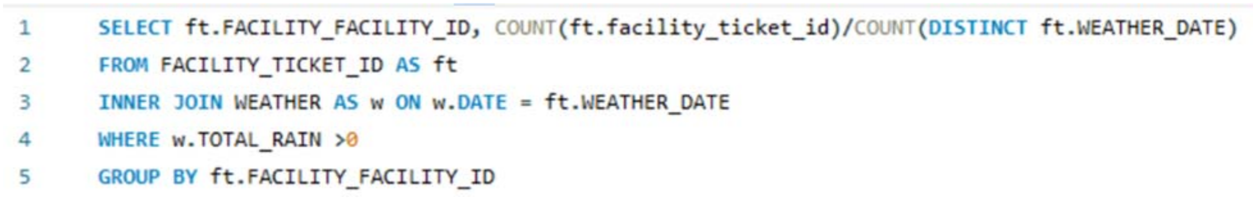
RESULTS (partial):



RIDE_RIDE_ID	count_spring	count_summer	count_fall	count_winter
R001	33	21	6	34
R002	36	24	11	26
R003	38	25	9	32
R004	33	24	8	34
R005	47	21	10	45
R006	34	30	6	31
R007	36	20	13	38
R008	39	27	6	33
R009	29	24	4	30
R010	46	23	4	34
R011	20	18	7	25

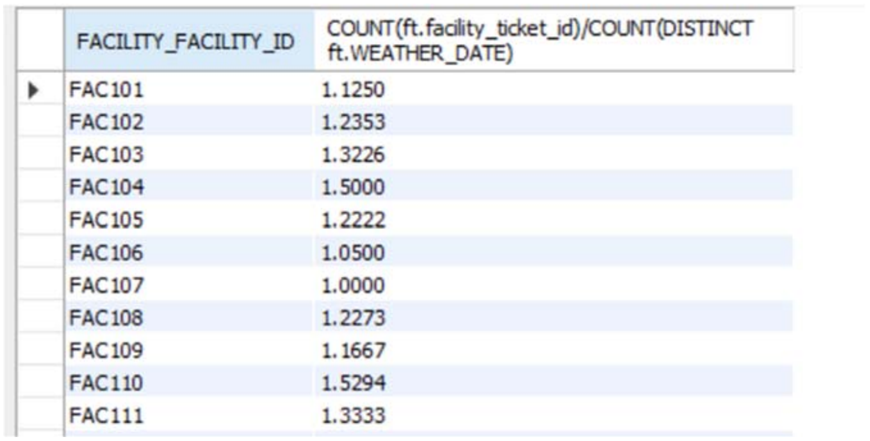
Q3. Find the average number of visits per day for each facility when raining

Codes:



```sql
SELECT ft.FACILITY_FACILITY_ID, COUNT(ft.facility_ticket_id)/COUNT(DISTINCT ft.WEATHER_DATE)
FROM FACILITY_TICKET_ID AS ft
INNER JOIN WEATHER AS w ON w.DATE = ft.WEATHER_DATE
WHERE w.TOTAL_RAIN >0
GROUP BY ft.FACILITY_FACILITY_ID
```

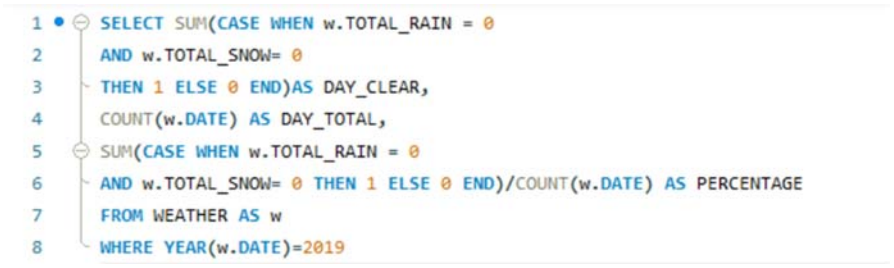
Results (partial):



FACILITY_FACILITY_ID	COUNT(ft.facility_ticket_id)/COUNT(DISTINCT ft.WEATHER_DATE)
FAC101	1.1250
FAC102	1.2353
FAC103	1.3226
FAC104	1.5000
FAC105	1.2222
FAC106	1.0500
FAC107	1.0000
FAC108	1.2273
FAC109	1.1667
FAC110	1.5294
FAC111	1.3333

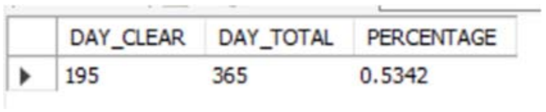
Q4. Find how many days are there in 2019 where there is no snow and there is no rain

Codes:



```sql
SELECT SUM(CASE WHEN w.TOTAL_RAIN = 0
AND w.TOTAL_SNOW= 0
THEN 1 ELSE 0 END)AS DAY_CLEAR,
COUNT(w.DATE) AS DAY_TOTAL,
SUM(CASE WHEN w.TOTAL_RAIN = 0
AND w.TOTAL_SNOW= 0 THEN 1 ELSE 0 END)/COUNT(w.DATE) AS PERCENTAGE
FROM WEATHER AS w
WHERE YEAR(w.DATE)=2019
```

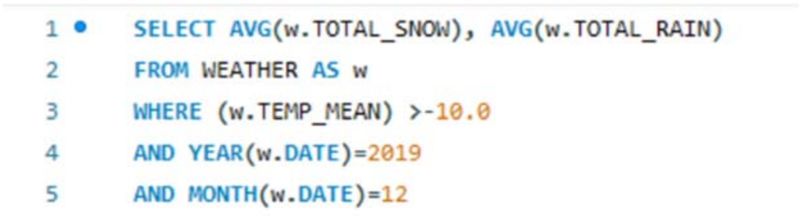
Results:



DAY_CLEAR	DAY_TOTAL	PERCENTAGE
195	365	0.5342

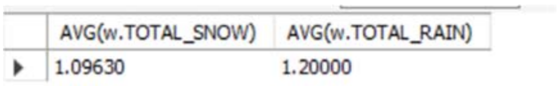
Q5. Find the average amount of snow and rains in 2019 in december where the temperature is above -10 degree celsius.

Codes:



```sql
SELECT AVG(w.TOTAL_SNOW), AVG(w.TOTAL_RAIN)
FROM WEATHER AS w
WHERE (w.TEMP_MEAN) >-10.0
AND YEAR(w.DATE)=2019
AND MONTH(w.DATE)=12
```

Results:



AVG(w.TOTAL_SNOW)	AVG(w.TOTAL_RAIN)
1.09630	1.20000