NetID: jc12300

Name: Jie Cheng

Course: CSGY-6083-Principles of Database Systems

Section: B

Date of submission: Nov 25, 2023

CS-GY 6083 - B, Fall 2023
Principles of Database Systems

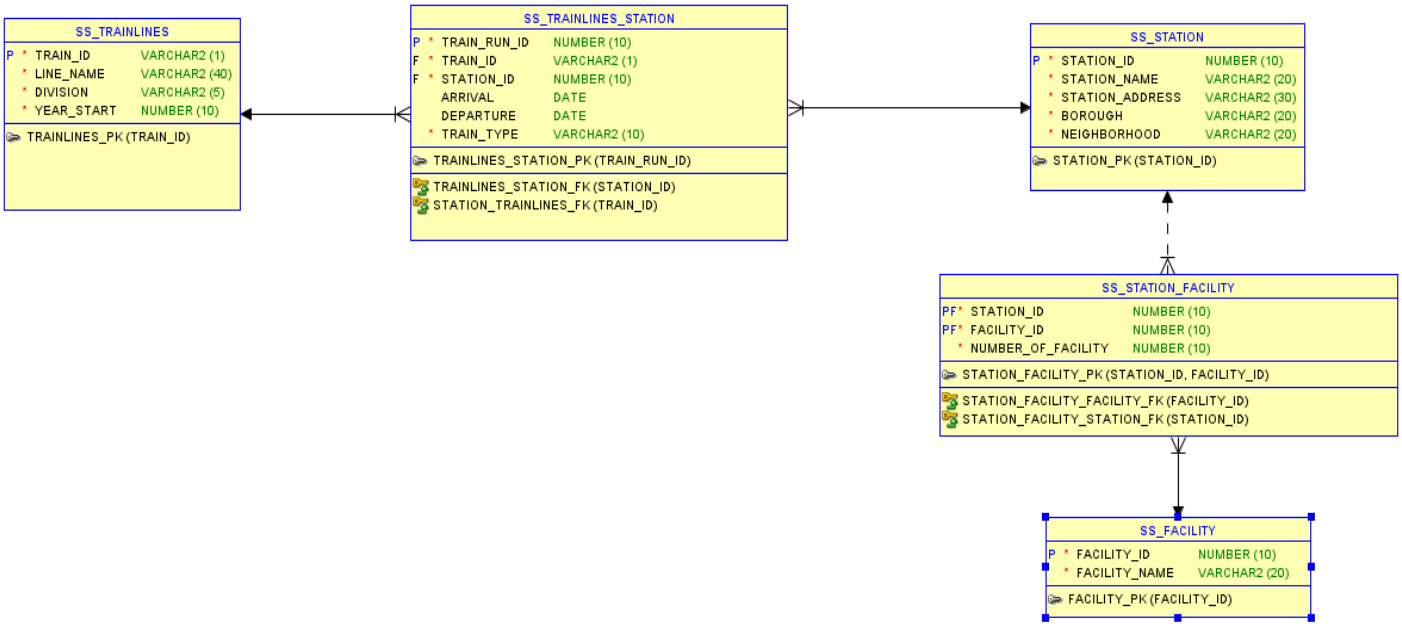
ASSIGNMENT: 3 [100 Points, 5% weight to the Final grades]

Please submit your assignment on NYU Brightspace in a single PDF document attachment. Please mention Student ID, Name, Course, Section Number, and date of submission on first page of your submission. Please use SQL file (DDL and DML code) for writing SQLs in Oracle. You may create equivalent DDL and DML code for MySQL and write SQLs in MySQL. All the tables should be created with your initial as a prefix. DO NOT change data in tables. You can write SQL in either Oracle or MySQL.

Problem A: 50 points [each question has 10 points]

The New York City Subway is a rapid transit system in the New York City boroughs of Manhattan, Brooklyn, Queens, and the Bronx. It is owned by the government of New York City and leased to the New York City Transit Authority, an affiliate agency of the state-run Metropolitan Transportation Authority (MTA). Opened on October 27, 1904, the New York City Subway is one of the world's oldest public transit systems, one of the most used, and the one with the most stations, with 472 stations in operation and 36 train lines.

SAMTA (Subway Administration of Metropolitan Transportation Authority) is the NYU affiliated data analytics startup company. SAMTA has undertaken a database project to normalize subway data systems. For relational schema model in figure, write SQL statements to answer following questions. <u>You don't need to create a relational model.</u>



i. **List all the details of the station which has the highest number of facilities in Elevator.**

```sql
SELECT s.*
FROM jc_station s
JOIN (
    SELECT station_id
    FROM jc_station_facility
    WHERE facility_id = (SELECT facility_id FROM jc_facility WHERE facility_name = 'Elevator')
    ORDER BY number_of_facility DESC
    FETCH FIRST ROW ONLY
) sf ON s.station_id = sf.station_id;
```

STATION_ID	STATION_NAME	STATION_ADDRESS	BOROUGH	NEIGHBORHOOD
1	Times Sq - 42nd St	234 W 42nd St	Manhattan	Times Square

ii. **List details of each train line and their highest run time. Your result should have TRAIN_ID, LINE_NAME, ARRIVVAL, DEPARTURE,**

```sql
WITH MaxRunTimePerLine AS (
    SELECT
        t.train_id,
        t.line_name,
        MAX(ts.departure - ts.arrival) AS max_run_time
    FROM
        jc_trainlines t
        JOIN jc_trainlines_station ts ON t.train_id = ts.train_id
    GROUP BY
        t.train_id, t.line_name
)
SELECT
    m.train_id,
    m.line_name,
    TO_CHAR(ts.arrival, 'YYYY-MM-DD HH24:MI:SS') AS arrival,
    TO_CHAR(ts.departure, 'YYYY-MM-DD HH24:MI:SS') AS departure
FROM
    MaxRunTimePerLine m
    JOIN jc_trainlines_station ts ON m.train_id = ts.train_id
WHERE
    ts.departure - ts.arrival = m.max_run_time;
```

TRAIN_ID	LINE_NAME	ARRIVAL	DEPARTURE
N	Broadway Express	2023-10-14 18:00:00	2023-10-14 18:37:00
Q	Second Avenue/Broadway Express	2023-10-14 19:00:00	2023-10-14 19:35:23
W	Broadway Local	2023-10-14 21:00:00	2023-10-14 21:33:36
A	Eighth Avenue Express	2023-10-15 08:00:00	2023-10-15 08:33:34
B	Sixth Avenue Express	2023-10-15 09:00:00	2023-10-15 09:34:38
C	Eighth Avenue Local	2023-10-15 10:00:00	2023-10-15 10:32:39
D	Sixth Avenue Express	2023-10-15 11:00:00	2023-10-15 11:34:12
E	Eighth Avenue Local	2023-10-15 12:00:00	2023-10-15 12:31:55
F	Queens Boulevard Express	2023-10-15 13:00:00	2023-10-15 13:33:19
G	Brooklyn-Queens Crosstown	2023-10-15 14:00:00	2023-10-15 14:33:27
J	Nassau Street Express	2023-10-15 15:00:00	2023-10-15 15:31:58
L	Fourteenth Street-Canarsie	2023-10-15 16:00:00	2023-10-15 16:35:09
M	Queens Boulevard Local	2023-10-15 17:00:00	2023-10-15 17:36:03
R	Broadway Local	2023-10-15 20:00:00	2023-10-15 20:34:12
Z	Nassau Street Local	2023-10-15 22:00:00	2023-10-15 22:32:48

Download CSV

iii. **Find the top two boroughs in terms of numbers of station facilities. Your result should have the name of the borough and number of facilities.**

```sql
SELECT
    s.borough,
    SUM(sf.number_of_facility) AS total_facilities
FROM jc_station_facility sf
JOIN jc_station s ON sf.station_id = s.station_id
GROUP BY s.borough
ORDER BY total_facilities DESC
FETCH FIRST 2 ROWS ONLY;
```

BOROUGH	TOTAL_FACILITIES
Manhattan	52
Queens	5

Download CSV

iv. **List station name, borough, neighborhood, facility name, and number of facilities. Arrange the result in descending order of number of facilities.**

```sql
SELECT
    s.station_name,
    s.borough,
    s.neighborhood,
    f.facility_name,
    sf.number_of_facility
FROM jc_station_facility sf
JOIN jc_station s ON sf.station_id = s.station_id
JOIN jc_facility f ON sf.facility_id = f.facility_id
ORDER BY sf.number_of_facility DESC;
```

STATION_NAME	BOROUGH	NEIGHBORHOOD	FACILITY_NAME	NUMBER_OF_FACILITY
Times Sq - 42nd St	Manhattan	Times Square	Restroom	6
Times Sq - 42nd St	Manhattan	Times Square	Bus Connection	6
Grand Central	Manhattan	Midtown East	Escalator	5
Times Sq - 42nd St	Manhattan	Times Square	Escalator	4
Times Sq - 42nd St	Manhattan	Times Square	Transfer Connection	4
Brooklyn Bridge	Manhattan	Civic Center	Restroom	4
Grand Central	Manhattan	Midtown East	Restroom	3
Union Square	Manhattan	Union Square	Restroom	3
Times Sq - 42nd St	Manhattan	Times Square	Elevator	3
Times Sq - 42nd St	Manhattan	Times Square	Wheelchair	2
Jamaica Center	Queens	Jamaica	Escalator	2
Jamaica Center	Queens	Jamaica	24 Hr Booth	2
Times Sq - 42nd St	Manhattan	Times Square	24 Hr Booth	2
Grand Central	Manhattan	Midtown East	Wheelchair	2
Union Square	Manhattan	Union Square	Transfer Connection	2
Brooklyn Bridge	Manhattan	Civic Center	Elevator	2
Jamaica Center	Queens	Jamaica	Wheelchair	1
Brooklyn Bridge	Manhattan	Civic Center	24 Hr Booth	1
Brooklyn Bridge	Manhattan	Civic Center	Escalator	1
Union Square	Manhattan	Union Square	Elevator	1
Brooklyn Bridge	Manhattan	Civic Center	Transfer Connection	1
Jamaica Center	Queens	Jamaica	Bus Connection	0
Jamaica Center	Queens	Jamaica	Transfer Connection	0
Jamaica Center	Queens	Jamaica	Restroom	0
Jamaica Center	Queens	Jamaica	Elevator	0
Brooklyn Bridge	Manhattan	Civic Center	Bus Connection	0
Brooklyn Bridge	Manhattan	Civic Center	Wheelchair	0
Union Square	Manhattan	Union Square	24 Hr Booth	0
Union Square	Manhattan	Union Square	Bus Connection	0
Union Square	Manhattan	Union Square	Escalator	0
Grand Central	Manhattan	Midtown East	24 Hr Booth	0
Grand Central	Manhattan	Midtown East	Bus Connection	0
Grand Central	Manhattan	Midtown East	Transfer Connection	0
Grand Central	Manhattan	Midtown East	Elevator	0
Union Square	Manhattan	Union Square	Wheelchair	0

Download CSV

35 rows selected.

v. **List the train line that has maximum number of stations.**

```
SELECT
    train_id,
    COUNT(*) AS station_count
FROM
    jc_trainlines_station
GROUP BY
    train_id
ORDER BY
    station_count DESC
FETCH FIRST 1 ROW ONLY;
```

TRAIN_ID	STATION_COUNT
A	3

Submit: For each of questions submit a) SQL b) SQL result. All SQL and corresponding results must be visible clearly on screenshots.

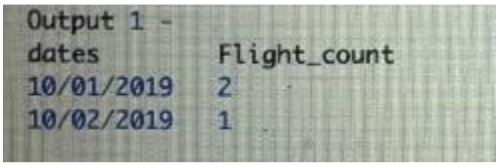
Problem 2: 20 points [each question has 10 points]

FLIGHT_ID	NODE_ID	STATUS	SCHEDULE
A123	SEA	DEP	10/01/2019 07:00:00
A123	MIA	ARR	10/01/2019 11:00:00
A123	MIA	DEP	10/02/2019 08:00:00
A123	LAX	ARR	10/01/2019 12:00:00
A234	SEA	DEP	10/01/2019 11:00:00
A234	MIA	ARR	10/01/2019 14:00:00

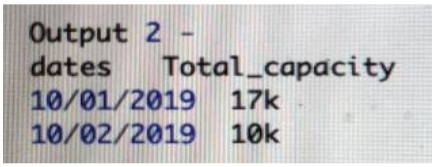
This is a FLIGHT table data for an Airline operator. NODE_ID represents Airport Code, and STATUS represents DEP (Departure) or ARR (Arrival)

FLIGHT_ID	FLIGHT_TYPE	CAPACITY
A123	767	10000
A234	737	7000

This is the SIZE table data that represent total CAPACITY of each FLIGHT_ID and FLIGHT_TYPE



Q1: Write a SQL query to find total number of unique flights that operated on each day. Your output result should appear as shown as picture Output 1.



Q2: Write a SQL query to find the total capacity for each day. Your output result should appear as shown in picture Output 2.

Submission:

Create tables as above with your initial as prefix such as AP_FLIGHT, AP_SIZE. Populate the same data as shown in pictures. For Q1 and Q2 write the SQLs that produce the desired results as Output 1 and Output 2. Submit screenshots of your SQL queries and their respective results. You can use Oracle or MySQL relational database. All SQL and corresponding results must be visible clearly on screenshots.

Q1

```
1 v SELECT TO_CHAR(schedule, 'MM/DD/YYYY') AS dates, COUNT(DISTINCT flight_id) AS Flight_count
2   FROM jc_flight
3   GROUP BY TO_CHAR(schedule, 'MM/DD/YYYY')
4   ORDER BY dates;
```

DATES	FLIGHT_COUNT
10/01/2019	2
10/02/2019	1

Download CSV

Q2

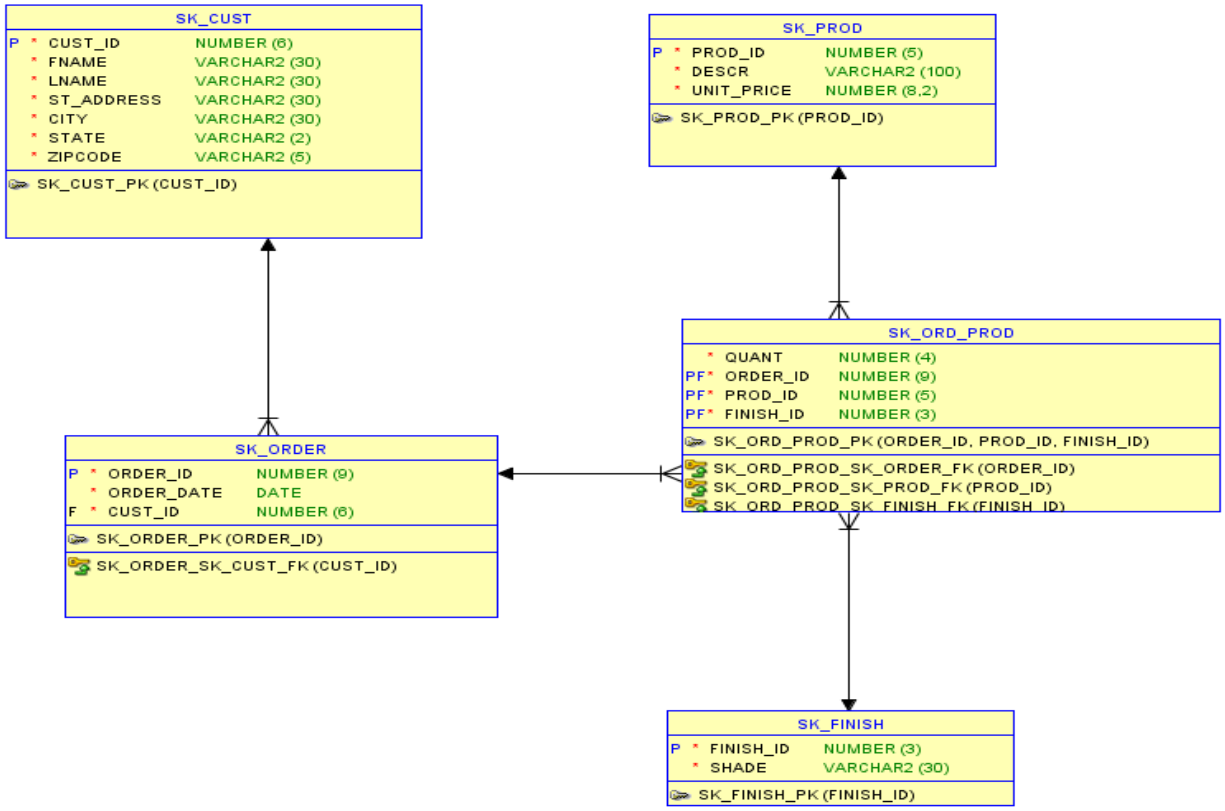
```
SELECT
    dates AS dates, TO_CHAR(SUM(capacity) / 1000, '999,999') || 'K' AS Total_capacity
FROM
   (SELECT DISTINCT(flight_id), TO_CHAR(schedule, 'MM/DD/YYYY') AS dates FROM jc_flight) jf
JOIN
    jc_size js ON jf.flight_id = js.flight_id
GROUP BY
    dates
ORDER BY
    dates;
```

DATES	TOTAL_CAPACITY
10/01/2019	17K
10/02/2019	10K

Download CSV

Problem 3: 30 points

For a given relational model below, please find attached file containing DDL and DMLs. You may create equivalent code for MySQL. Create tables and insert data by replacing SK with your own initial. You can do this assignment either in Oracle or MySQL. <u>You don't need to draw logical/relational models.</u>



I. For this relational model of a furniture company, create a read-only database view that represents the following dataset. Customer ID, Customer Name (both First and Last name), Order_Id, order date, each product in order with description, quantity, Unit_Price, Total price of each product, and Finish shade. Sort the dataset in order of total order amount. Give appropriate column names in view. Restrict dataset to represent only those orders which have total value over $1000. Once view is created submit View code and then retrieve result of view using SQL query.

```
CREATE VIEW order_details_view AS
SELECT
    c.cust_id,
    c.fname || ' ' || c.lname AS customer_name,
    o.order_id,
    o.order_date,
    p.prod_id,
    p.descr AS product_description,
    op.quant AS quantity,
    p.unit_price,
    (op.quant * p.unit_price) AS total_price,
    f.shade AS finish_shade
FROM
    jc_cust c
JOIN
    jc_order o ON c.cust_id = o.cust_id
JOIN
    jc_ord_prod op ON o.order_id = op.order_id
JOIN
    jc_prod p ON op.prod_id = p.prod_id
JOIN
    jc_finish f ON op.finish_id = f.finish_id
WHERE
    (op.quant * p.unit_price) > 1000
ORDER BY
    (op.quant * p.unit_price) DESC
WITH  READ ONLY;
```

CUST_ID	CUSTOMER_NAME	ORDER_ID	ORDER_DATE	PROD_ID	PRODUCT_DESCRIPTION	QUANTITY	UNIT_PRICE	TOTAL_PRICE	FINISH_SHADE
22334	William Martinez	1023	03-JUL-17	13	Armoire	2	1500	3000	Simply White
45678	John Miller	1022	10-JUN-18	10	Bookcase	3	1000	3000	Cherry
90876	Robert Jones	1010	24-FEB-20	10	Bookcase	2	1000	2000	Cherry
99876	Linda Wilson	1012	18-SEP-18	8	Wardrobe	2	900	1800	Classic Gray
99876	Linda Wilson	1024	05-MAY-18	14	Windsor Chair	2	890	1780	Natural
10987	Karen Iglesias	1009	12-APR-21	13	Armoire	1	1500	1500	Fruitwood
67890	Mary Brown	1020	09-JUL-20	3	Couch	2	670	1340	Provincial
45678	John Miller	1022	10-JUN-18	4	Entertainment Center	2	650	1300	Red Chestnut
98987	Richard Martin	1015	20-FEB-17	2	Cabinets	2	525	1050	Natural Ash

II. Find top 3 products and their finish shade in terms of total quantities sold between 01/01/2018 and 12/31/2021. Your result dataset should have Product_Id, Product Description, shade, and total quantity sold.

```
SELECT
    p.prod_id,
    p.descr AS product_description,
    f.shade AS finish_shade,
    SUM(op.quant) AS total_quantity_sold
FROM
    jc_ord_prod op
JOIN
    jc_prod p ON op.prod_id = p.prod_id
JOIN
    jc_finish f ON op.finish_id = f.finish_id
JOIN
    jc_order o ON op.order_id = o.order_id
WHERE
    o.order_date BETWEEN TO_DATE('01-01-2018', 'MM-DD-YYYY') AND TO_DATE('12-31-2021', 'MM-DD-YYYY')
GROUP BY
    p.prod_id, p.descr, f.shade
ORDER BY
    total_quantity_sold DESC
FETCH FIRST 3 ROWS ONLY;
```

PROD_ID	PRODUCT_DESCRIPTION	FINISH_SHADE	TOTAL_QUANTITY_SOLD
10	Bookcase	Cherry	5
9	Washstand	Simply White	3
11	4-Dr Dresser	Oak	2

III. Find products and their finish shade that have not been sold in during the months of Oct, Nov, and Dec in 2020

```sql
SELECT
    p.prod_id,
    p.descr AS product_description,
    f.shade AS finish_shade
FROM
    jc_ord_prod op
JOIN
    jc_prod p ON op.prod_id = p.prod_id
JOIN
    jc_finish f ON op.finish_id = f.finish_id
WHERE
    NOT EXISTS (
        SELECT 1
        FROM jc_ord_prod op
        JOIN jc_order o ON op.order_id = o.order_id
        WHERE p.prod_id = op.prod_id
          AND f.finish_id = op.finish_id
          AND EXTRACT(MONTH FROM o.order_date) IN (10, 11, 12)
          AND EXTRACT(YEAR FROM o.order_date) = 2020
    );
```

PROD_ID	PRODUCT_DESCRIPTION	FINISH_SHADE
4	Entertainment Center	Red Chestnut
9	Washstand	Simply White
13	Armoire	Fruitwood
10	Bookcase	Cherry
10	Bookcase	Cherry
15	Nightstand	Puritan Prine
15	Nightstand	Fruitwood
3	Couch	Provincial
7	Dinning Table	Natural Ash
1	Cradle Bed	Driftwood
5	Writer Desk	Natural
10	Bookcase	Red Chestnut
4	Entertainment Center	Green Velvet
6	Settee	Fruitwood
9	Washstand	Gunstock
9	Washstand	Green Velvet
4	Entertainment Center	Cherry
8	Wardrobe	Classic Gray
8	Wardrobe	Green Velvet
14	Windsor Chair	Natural
1	Cradle Bed	Puritan Prine
13	Armoire	Simply White
5	Writer Desk	Driftwood

Submission:

For each question write an SQL query and submit both SQL and a clearly visible screenshot of corresponding result underneath each question. Make appropriate use of column alias and built in functions in your SQL queries.