**Database System Final Project Report**

“Quantum Flux - TCG SHOP”



Team Member:

Muhammad Alif Satriadhi 5025221188

Fawwas Aldy Nurramdhan Kaisar 5025221179

Mohmmad Hanif Furqan Aufa Putra 5025221161

Surya Fadli Alamsyah 5025221059

Lecturer:

Ratih Nur Esti Anggraini, S.Kom, M.Sc., Ph.D.

INSTITUT TEKNOLOGI SEPULUH NOPEMBER

SURABAYA

2023/2024

1. Initial Scenario

Given the current boom in Trading Card Games (TCG), an entrepreneur who is also a fan of TCG has decided to open a TCG-themed Game Centre. Recognizing the need for a streamlined transaction process, the entrepreneur has enlisted the services of an acquaintance to design the database system. The Game Centre will offer various card packs from different series (Pokémon, Yu-Gi-Oh, Magic the Gathering, etc.) and feature a card collection display, with some cards available for sale. Additionally, it also offers Game Boards. The owner is a geek, so he is also a veteran player, so he also became a card collector. Therefore, some of the cards owned by the owner are also sold to players. Players can buy/sell/trade cards with the owner's card collection.

The Game Centre also offers a TCG game area where players can play against one another, it also offers a private room if the player doesn't want to wait their turn to play. The private game rooms are divided into three types (Regular, Deluxe, and VIP).  To become a member, the player needs to have provide personal data, including their name and email. The benefits of becoming a member are Access to play in the Game Area, earning points with every transaction, which can be exchanged for merchandise, and Enjoying discounts.

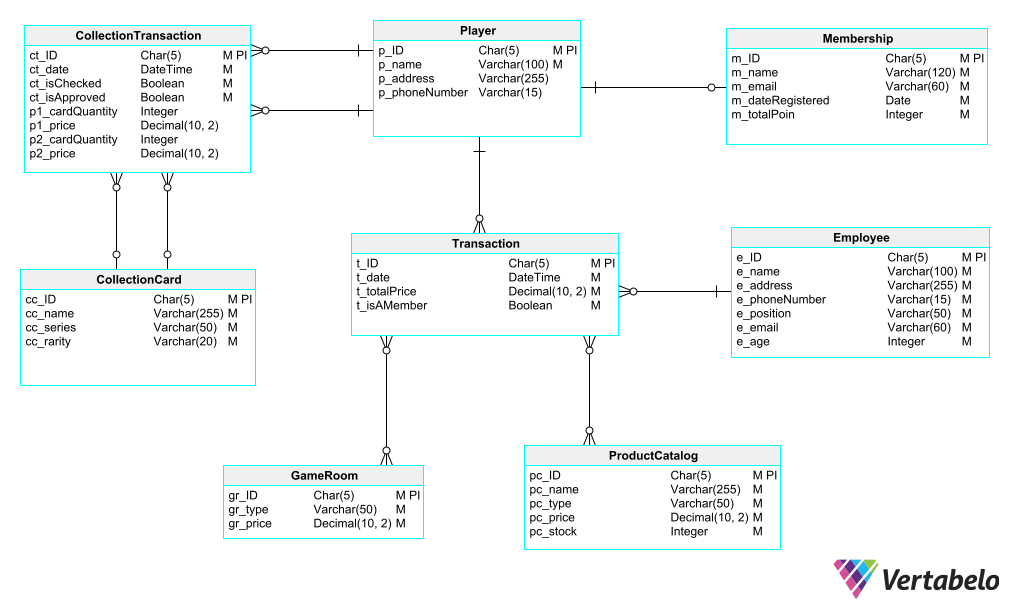
The entrepreneur desires a system that records employee data, including Employee ID, name, gender, phone number, email, and age. Each employee can handle multiple transactions by customers. Customers have the option to purchase more than one item per transaction. Employees will record whether the customer is a member, the total number of items purchased, the total amount due, and the preferred payment method.

The database design aims to facilitate efficient and accurate record-keeping for both employee management and customer transactions, ensuring a seamless experience for all patrons of the TCG-themed Game Centre.

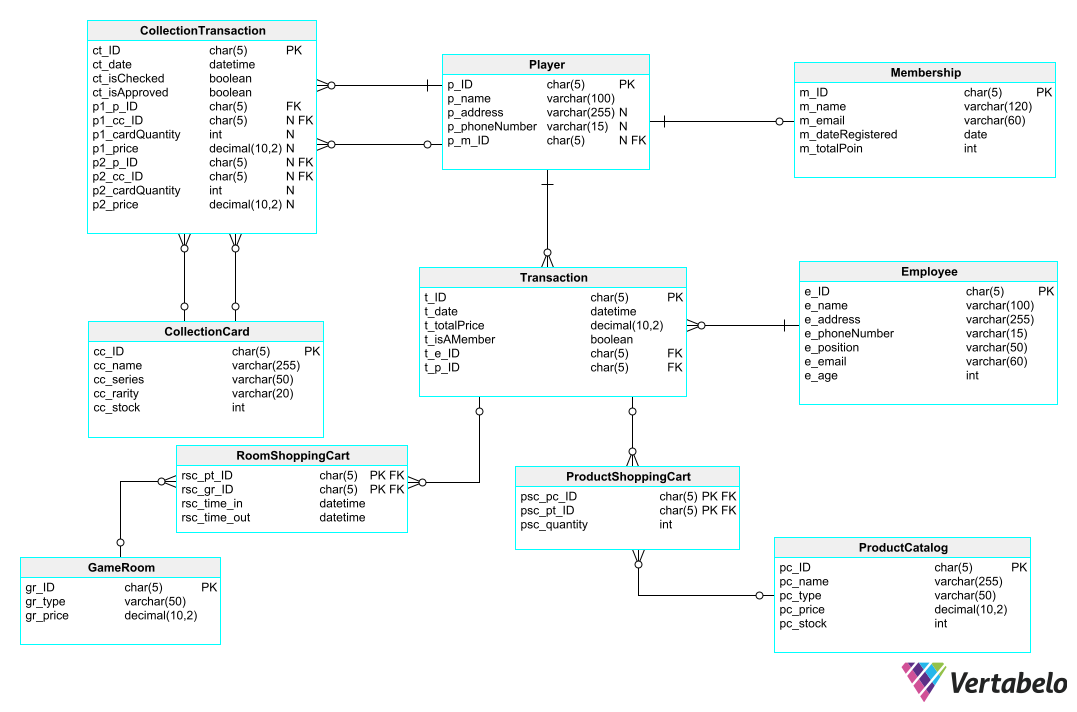
**Assumptions:**

* The Owner is included as the player with a unique ID “OWNER” in CollectionCardTransaction

1. CDM



1. PDM



1. DDL

CREATE TABLE CollectionCard (

    cc\_ID char(5) NOT NULL,

    cc\_name varchar(255) NOT NULL,

    cc\_series varchar(50) NOT NULL,

    cc\_rarity varchar(20) NOT NULL,

    CONSTRAINT CollectionCard\_pk PRIMARY KEY (cc\_ID)

);

CREATE TABLE CollectionTransaction (

    ct\_ID char(5) NOT NULL,

    ct\_date datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP,

    ct\_isChecked boolean NOT NULL DEFAULT false,

    ct\_isApproved boolean NOT NULL DEFAULT false,

    p1\_p\_ID char(5) NOT NULL,

    p1\_cc\_ID char(5) NULL,

    p1\_cardQuantity int NULL,

    p1\_price decimal(10, 2) NULL,

    p2\_p\_ID char(5) NULL,

    p2\_cc\_ID char(5) NULL,

    p2\_cardQuantity int NULL,

    p2\_price decimal(10, 2) NULL,

    CONSTRAINT CollectionTransaction\_pk PRIMARY KEY (ct\_ID)

);

CREATE TABLE Employee (

    e\_ID char(5) NOT NULL,

    e\_name varchar(100) NOT NULL,

    e\_address varchar(255) NOT NULL,

    e\_phoneNumber varchar(15) NOT NULL,

    e\_position varchar(50) NOT NULL,

    e\_email varchar(60) NOT NULL,

    e\_age int NOT NULL,

    CONSTRAINT Employee\_pk PRIMARY KEY (e\_ID)

);

CREATE TABLE GameRoom (

    gr\_ID char(5) NOT NULL,

    gr\_type varchar(50) NOT NULL,

    gr\_price decimal(10, 2) NOT NULL,

    CONSTRAINT GameRoom\_pk PRIMARY KEY (gr\_ID)

);

CREATE TABLE Membership (

    m\_ID char(5) NOT NULL,

    m\_name varchar(120) NOT NULL,

    m\_email varchar(60) NOT NULL,

    m\_dateRegistered date NOT NULL DEFAULT CURRENT\_TIMESTAMP,

    m\_totalPoin int NOT NULL DEFAULT 0 CHECK (m\_totalPoin >= 0),

    CONSTRAINT Membership\_pk PRIMARY KEY (m\_ID)

);

CREATE TABLE Player (

    p\_ID char(5) NOT NULL,

    p\_name varchar(100) NOT NULL,

    p\_address varchar(255) NULL,

    p\_phoneNumber varchar(15) NULL,

    p\_m\_ID char(5) NULL,

    CONSTRAINT Player\_pk PRIMARY KEY (p\_ID)

);

CREATE TABLE ProductCatalog (

    pc\_ID char(5) NOT NULL,

    pc\_name varchar(255) NOT NULL,

    pc\_type varchar(50) NOT NULL,

    pc\_price decimal(10, 2) NOT NULL,

    pc\_stock int NOT NULL DEFAULT 0 CHECK (pc\_stock >= 0),

    CONSTRAINT ProductCatalog\_pk PRIMARY KEY (pc\_ID)

);

CREATE TABLE ProductShoppingCart (

    psc\_pc\_ID char(5) NOT NULL,

    psc\_pt\_ID char(5) NOT NULL,

    psc\_quantity int NOT NULL CHECK (psc\_quantity >= 0),

    CONSTRAINT ProductCatalog\_Transaction\_pk PRIMARY KEY (psc\_pc\_ID, psc\_pt\_ID)

);

CREATE TABLE RoomShoppingCart (

    rsc\_pt\_ID char(5) NOT NULL,

    rsc\_gr\_ID char(5) NOT NULL,

    rsc\_time\_in datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP,

    rsc\_time\_out datetime NOT NULL DEFAULT (CURRENT\_TIMESTAMP + INTERVAL 1 HOUR) CHECK (rsc\_time\_out > rsc\_time\_in),

    CONSTRAINT Transaction\_GameRoom\_pk PRIMARY KEY (rsc\_pt\_ID, rsc\_gr\_ID)

);

CREATE TABLE Transaction (

    t\_ID char(5) NOT NULL,

    t\_date datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP,

    t\_totalPrice decimal(10, 2) NOT NULL DEFAULT 0 CHECK (t\_totalPrice >= 0),

    t\_isAMember boolean NOT NULL DEFAULT false,

    t\_e\_ID char(5) NOT NULL,

    t\_p\_ID char(5) NOT NULL,

    CONSTRAINT Transaction\_pk PRIMARY KEY (t\_ID)

);

ALTER TABLE

    CollectionTransaction

ADD

    CONSTRAINT CollectionCard\_CollectionTransaction\_P1 FOREIGN KEY CollectionCard\_CollectionTransaction\_P1 (p1\_cc\_ID) REFERENCES CollectionCard (cc\_ID);

ALTER TABLE

    CollectionTransaction

ADD

    CONSTRAINT CollectionTransaction\_CollectionCard\_P2 FOREIGN KEY CollectionTransaction\_CollectionCard\_P2 (p2\_cc\_ID) REFERENCES CollectionCard (cc\_ID);

ALTER TABLE

    CollectionTransaction

ADD

    CONSTRAINT CollectionTransaction\_Player\_P1 FOREIGN KEY CollectionTransaction\_Player\_P1 (p1\_p\_ID) REFERENCES Player (p\_ID);

ALTER TABLE

    CollectionTransaction

ADD

    CONSTRAINT CollectionTransaction\_Player\_P2 FOREIGN KEY CollectionTransaction\_Player\_P2 (p2\_p\_ID) REFERENCES Player (p\_ID);

ALTER TABLE

    Player

ADD

    CONSTRAINT Player\_Membership FOREIGN KEY Player\_Membership (p\_m\_ID) REFERENCES Membership (m\_ID);

ALTER TABLE

    ProductShoppingCart

ADD

    CONSTRAINT ProductCatalog\_Transaction\_ProductCatalog FOREIGN KEY ProductCatalog\_Transaction\_ProductCatalog (psc\_pc\_ID) REFERENCES ProductCatalog (pc\_ID);

ALTER TABLE

    ProductShoppingCart

ADD

    CONSTRAINT ProductCatalog\_Transaction\_Transaction FOREIGN KEY ProductCatalog\_Transaction\_Transaction (psc\_pt\_ID) REFERENCES Transaction (t\_ID);

ALTER TABLE

    Transaction

ADD

    CONSTRAINT ProductTransaction\_Employee FOREIGN KEY ProductTransaction\_Employee (t\_e\_ID) REFERENCES Employee (e\_ID);

ALTER TABLE

    Transaction

ADD

    CONSTRAINT ProductTransaction\_Player FOREIGN KEY ProductTransaction\_Player (t\_p\_ID) REFERENCES Player (p\_ID);

ALTER TABLE

    RoomShoppingCart

ADD

    CONSTRAINT Transaction\_GameRoom\_GameRoom FOREIGN KEY Transaction\_GameRoom\_GameRoom (rsc\_gr\_ID) REFERENCES GameRoom (gr\_ID);

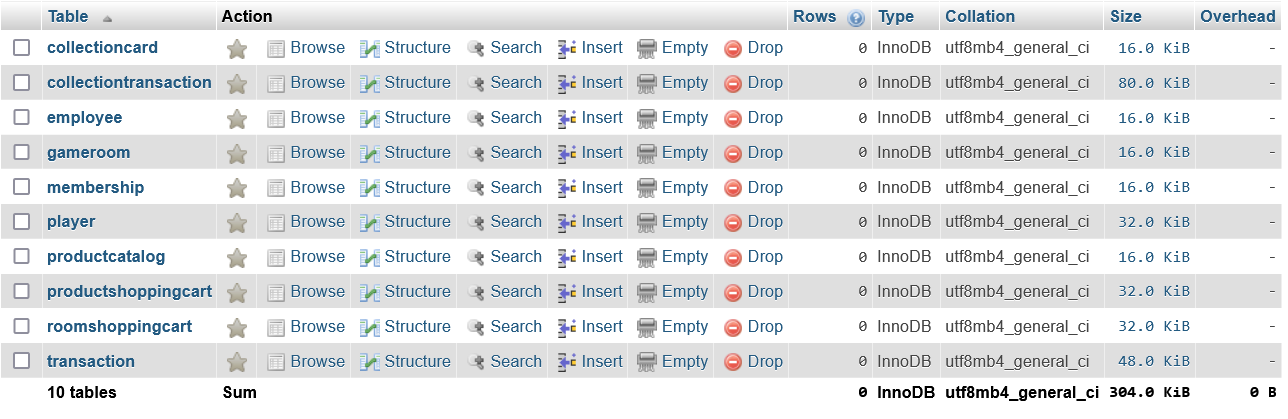
ALTER TABLE

    RoomShoppingCart

ADD

    CONSTRAINT Transaction\_GameRoom\_Transaction FOREIGN KEY Transaction\_GameRoom\_Transaction (rsc\_pt\_ID) REFERENCES Transaction (t\_ID);

Result:



The DML will be divided into 2 parts namely november and december parts followed by a query per part after the dml, this will allow a more realistic experience in operating the database.

5a. November’s DML

*-- Employee table*

INSERT INTO Employee (e\_ID, e\_name, e\_address, e\_phoneNumber, e\_position, e\_email, e\_age) VALUES

('E0001', 'Xiao Yuu', 'Jl. Jenderal Sudirman No. 123', '555-1111', 'Owner', 'xiaoyuu@gmail.com', 29),

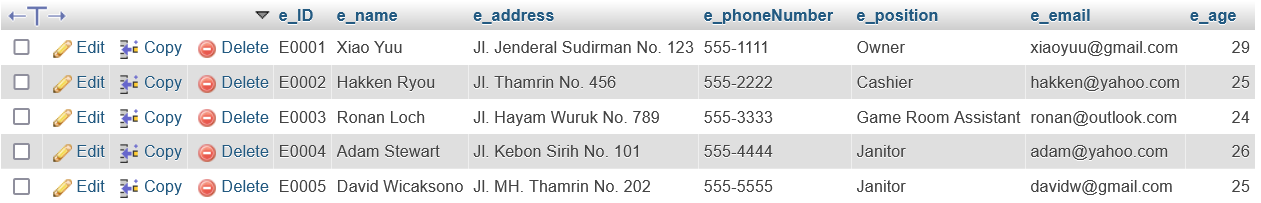
('E0002', 'Hakken Ryou', 'Jl. Thamrin No. 456', '555-2222', 'Cashier', 'hakken@yahoo.com', 25),

('E0003', 'Ronan Loch', 'Jl. Hayam Wuruk No. 789', '555-3333', 'Game Room Assistant', 'ronan@outlook.com', 24),

('E0004', 'Adam Stewart', 'Jl. Kebon Sirih No. 101', '555-4444', 'Janitor', 'adam@yahoo.com', 26),

('E0005', 'David Wicaksono', 'Jl. MH. Thamrin No. 202', '555-5555', 'Janitor', 'davidw@gmail.com', 25);

Result:



*-- ProductCatalog table*

INSERT INTO ProductCatalog (pc\_ID, pc\_name, pc\_type, pc\_price, pc\_stock) VALUES

('PC001', 'Vivid Voltage Booster Pack', 'Pokemon Booster Pack', 55000, 20),

('PC002', 'Scarlet & Violet Booster Pack', 'Pokemon Booster Pack', 65000, 10),

('PC003', 'Sword & Shield Booster Pack', 'Pokemon Booster Pack', 45000, 15),

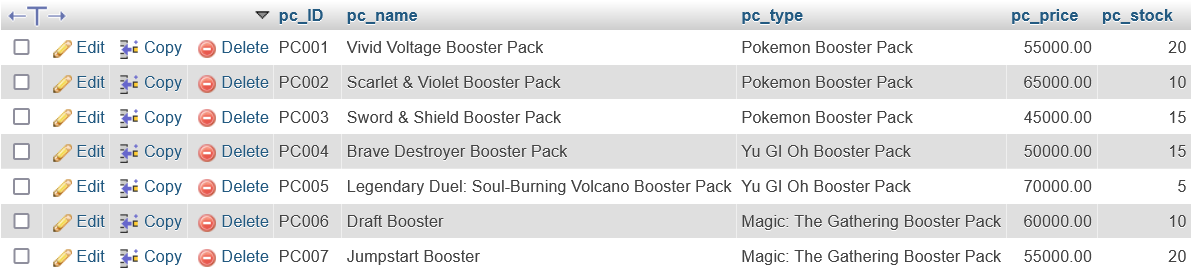
('PC004', 'Brave Destroyer Booster Pack', 'Yu GI Oh Booster Pack', 50000, 15),

('PC005', 'Legendary Duel: Soul-Burning Volcano Booster Pack', 'Yu GI Oh Booster Pack', 70000, 5),

('PC006', 'Draft Booster', 'Magic: The Gathering Booster Pack', 60000, 10),

('PC007', 'Jumpstart Booster', 'Magic: The Gathering Booster Pack', 55000, 20);

Result:



*-- GameRoom table*

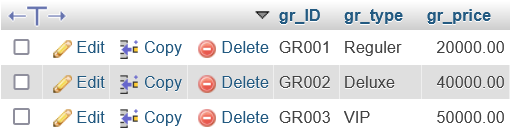
INSERT INTO GameRoom (gr\_ID, gr\_type, gr\_price) VALUES

('GR001', 'Reguler', 20000.00),

('GR002', 'Deluxe', 40000.00),

('GR003', 'VIP', 50000.00);

Result:



*-- Membership table*

INSERT INTO Membership (m\_ID, m\_name, m\_email, m\_dateRegistered, m\_totalPoin) VALUES

('OWNER', 'Master Rusdi', 'rusdi@gmail.com', '2023-01-01', 1000),

('M0001', 'Oda', 'oda@gmail.com', '2023-01-01', 100),

('M0002', 'Arsene', 'arsene@hotmail.com', '2023-02-15', 150),

('M0003', 'Rozen', 'rozen@yahoo.com', '2023-03-30', 200),

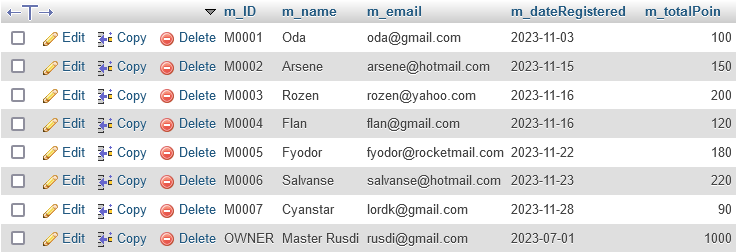
('M0004', 'Flan', 'flan@gmail.com', '2023-04-10', 120),

('M0005', 'Fyodor', 'fyodor@rocketmail.com', '2023-05-22', 180),

('M0006', 'Salvanse', 'salvanse@hotmail.com', '2023-06-05', 220),

('M0007', 'Cyanstar', 'lordk@gmail.com', '2023-07-15', 90);

Result:



In this data, Player Kang Ryu hasn’t registered as a member

*-- Player table*

INSERT INTO Player (p\_ID, p\_m\_ID, p\_name, p\_address, p\_phoneNumber) VALUES

('OWNER', 'OWNER', 'Rusdi', 'Jl. semangka', '111-111'),

('P0001', 'M0001', 'Teja Hoshizora', 'Jl. Merdeka No. 123', '555-1234'),

('P0002', 'M0002', 'Leo Leon', 'Jl. Kebon Sirih No. 456', '555-5678'),

('P0003', 'M0003', 'Alif Satria', 'Jl. Sudirman No. 789', '555-9012'),

('P0004', 'M0004', 'Rafli Rai', 'Jl. Pangeran Antasari No. 101', '555-3456'),

('P0005', 'M0005', 'Moris Idris', 'Jl. Gatot Subroto No. 202', '555-7890'),

('P0006', 'M0006', 'Hanif Kan', 'Jl. Thamrin No. 303', '555-1234'),

('P0007', 'M0007', 'Lord Kaisar', 'Jl. Hayam Wuruk No. 404', '555-5678'),

('P0008', NULL, 'Kang Ryu', 'Jl. Wahid Hasyim No. 505', '555-9012');

Result:



*-- CollectionCard table*

INSERT INTO CollectionCard (cc\_ID, cc\_name, cc\_series, cc\_rarity) VALUES

('CC001', 'Pikachu', 'Pokemon', 'Common'),

('CC002', 'Charizard', 'Pokemon', 'Rare'),

('CC003', 'Blastoise', 'Pokemon', 'Rare'),

('CC004', 'Dark Magician', 'Yu GI Oh', 'Ultra Rare'),

('CC005', 'Blue-Eyes White Dragon', 'Yu GI Oh', 'Ultra Rare'),

('CC006', 'Exodia the Forbidden One', 'Yu GI Oh', 'Secret Rare'),

('CC007', 'Jace, the Mind Sculptor', 'Magic: The Gathering', 'Mythic Rare'),

('CC008', 'Black Lotus', 'Magic: The Gathering', 'Mythic Rare'),

('CC009', 'Serra Angel', 'Magic: The Gathering', 'Rare'),

('CC010', 'Mewtwo', 'Pokemon', 'Legendary Rare'),

('CC011', 'Red-Eyes Black Dragon', 'Yu GI Oh', 'Super Rare'),

('CC012', 'Gyarados', 'Pokemon', 'Rare'),

('CC013', 'Force of Will', 'Magic: The Gathering', 'Rare'),

('CC014', 'Zygarde', 'Pokemon', 'Rare'),

('CC015', 'Raigeki', 'Yu GI Oh', 'Super Rare');

Result:



*-- Transaction table*

INSERT INTO Transaction (t\_ID, t\_date, t\_totalPrice, t\_isAMember, t\_e\_ID, t\_p\_ID) VALUES

('T0001', '2023-11-01 12:30:00', 0.00, 1, 'E0001', 'P0001'),

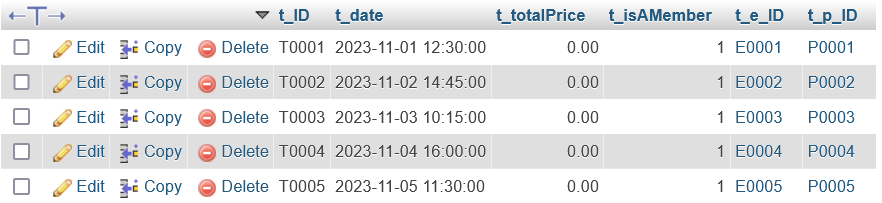
('T0002', '2023-11-02 14:45:00', 0.00, 1, 'E0002', 'P0002'),

('T0003', '2023-11-03 10:15:00', 0.00, 1, 'E0003', 'P0003'),

('T0004', '2023-11-04 16:00:00', 0.00, 1, 'E0004', 'P0004'),

('T0005', '2023-11-05 11:30:00', 0.00, 1, 'E0005', 'P0005');

Result:



*-- ProductShoppingCart table*

INSERT INTO ProductShoppingCart (psc\_pc\_ID, psc\_pt\_ID, psc\_quantity) VALUES

('PC002', 'T0001', 2),

('PC004', 'T0001', 1),

('PC005', 'T0002', 3),

('PC007', 'T0002', 1),

('PC003', 'T0003', 2),

('PC006', 'T0003', 1),

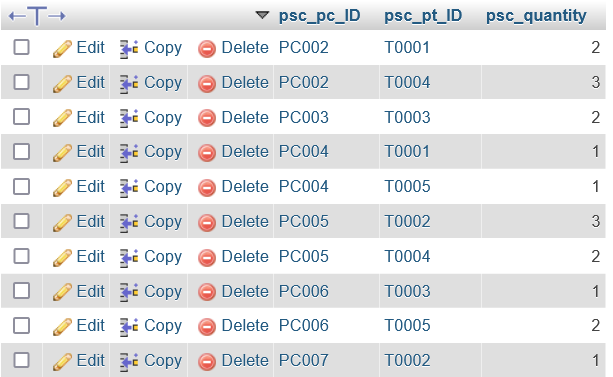
('PC002', 'T0004', 3),

('PC005', 'T0004', 2),

('PC004', 'T0005', 1),

('PC006', 'T0005', 2);

Result:



*-- RoomShoppingCart table*

INSERT INTO RoomShoppingCart (rsc\_pt\_ID, rsc\_gr\_ID, rsc\_time\_in, rsc\_time\_out) VALUES

('T0001', 'GR001', '2023-11-01 12:30:00', '2023-11-01 15:30:00'),

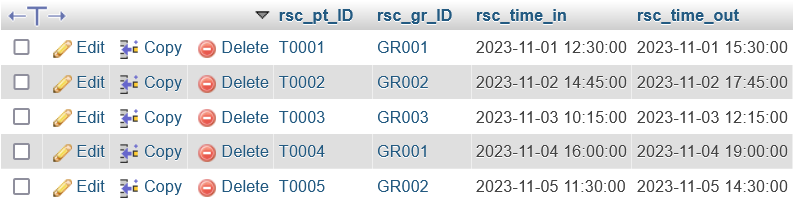
('T0002', 'GR002', '2023-11-02 14:45:00', '2023-11-02 17:45:00'),

('T0003', 'GR003', '2023-11-03 10:15:00', '2023-11-03 12:15:00'),

('T0004', 'GR001', '2023-11-04 16:00:00', '2023-11-04 19:00:00'),

('T0005', 'GR002', '2023-11-05 11:30:00', '2023-11-05 14:30:00');

Result:



*-- update transaction*

UPDATE Transaction

SET t\_totalPrice = (

    SELECT SUM(pc\_price \* psc\_quantity)

    FROM ProductShoppingCart

    JOIN ProductCatalog ON ProductShoppingCart.psc\_pc\_ID = ProductCatalog.pc\_ID

    WHERE ProductShoppingCart.psc\_pt\_ID = Transaction.t\_ID

) + (

    SELECT SUM(gr\_price \* TIMESTAMPDIFF(SECOND, rsc\_time\_in, rsc\_time\_out) / 3600)

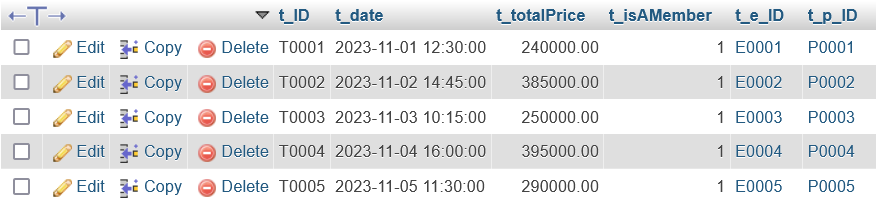
    FROM RoomShoppingCart

    JOIN GameRoom ON RoomShoppingCart.rsc\_gr\_ID = GameRoom.gr\_ID

    WHERE RoomShoppingCart.rsc\_pt\_ID = Transaction.t\_ID

)

Result:



*-- CollectionTransaction table*

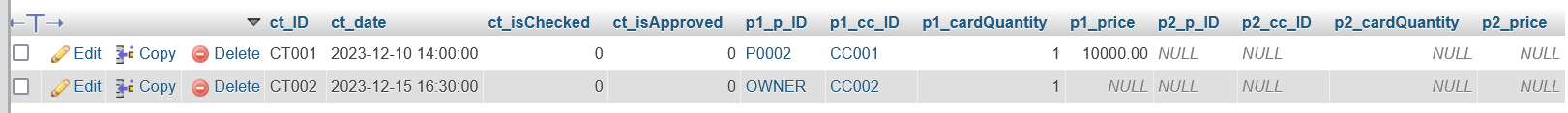
INSERT INTO CollectionTransaction

(ct\_ID, ct\_date, ct\_isChecked, ct\_isApproved, p1\_p\_ID, p1\_cc\_ID, p1\_cardQuantity, p1\_price, p2\_p\_ID, p2\_cc\_ID, p2\_cardQuantity, p2\_price) VALUES

('CT001', '2023-12-10 14:00:00', false, false, 'P0002', 'CC001', 1, 10000.00, NULL, NULL, NULL, NULL),

('CT002', '2023-12-15 16:30:00', false, false, 'OWNER', 'CC002', 1, NULL, NULL, NULL, NULL, NULL);

Result:



*-- Update for the first transaction (P1 Buying P2 card) & (Trading from P1 with P2 card but rejected)*

UPDATE CollectionTransaction

SET p2\_p\_ID = 'P0001'

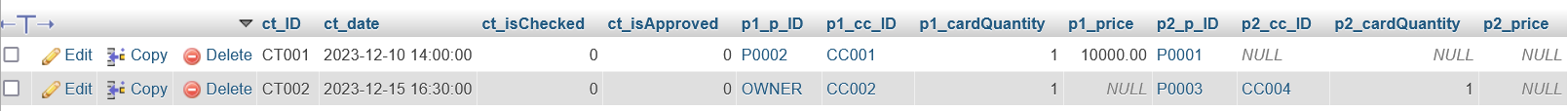
WHERE ct\_ID = 'CT001';

UPDATE CollectionTransaction

SET p2\_p\_ID = 'P0003', p2\_cc\_ID = 'CC004', p2\_cardQuantity = 1

WHERE ct\_ID = 'CT002';

Result:



*-- Update for the confirmation for each transaction*

UPDATE CollectionTransaction

SET ct\_isChecked = true,

    ct\_isApproved = true

WHERE ct\_ID = 'CT001';

UPDATE CollectionTransaction

SET ct\_isChecked = true,

    ct\_isApproved = false,

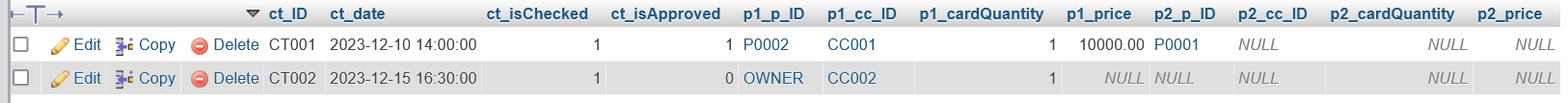
    p2\_p\_ID = NULL,

    p2\_cc\_ID = NULL,

    p2\_cardQuantity = NULL

WHERE ct\_ID = 'CT002';

Result:



6a. November’s Query

*-- Basic Query*

*-- 1 Tampilkan nama player dan memebrship point sort descending (sort)*

SELECT

    Player.p\_name AS PlayerName,

    Membership.m\_name AS MemberName,

    Membership.m\_totalPoin AS MembershipPoints

FROM

    Player

JOIN

    Membership ON Player.p\_m\_ID = Membership.m\_ID

ORDER BY

    Membership.m\_totalPoin DESC;

Result:



*-- 2 Tampilkan informasi transaksi koleksi kartu yang berkaitan dengan owner (filter)*

SELECT ct.\*, cc.\*,p.p\_name

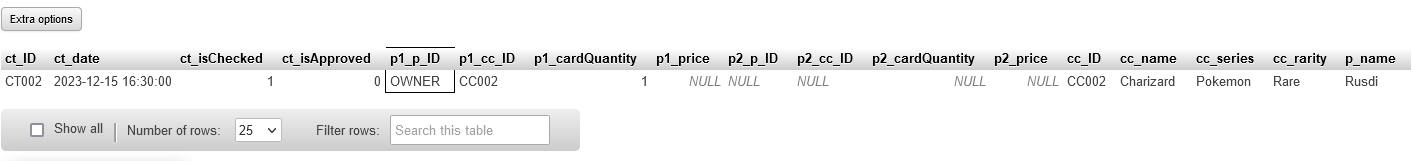
FROM collectiontransaction ct

JOIN collectioncard cc ON ct.p1\_cc\_ID = cc.cc\_ID

JOIN player p ON ct.p1\_p\_ID = p\_id

WHERE ct.p1\_p\_ID = 'OWNER';

Result:



*-- 3 Find most expensive product*

SELECT

    pc\_name AS MostExpensiveProduct,

    pc\_price AS MaxPrice

FROM

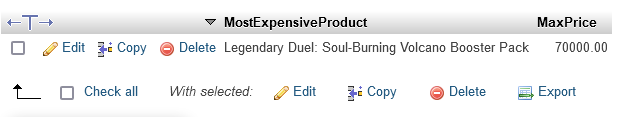
    ProductCatalog

ORDER BY

    pc\_price DESC

LIMIT 1;

Result:



*-- 4 Show collection card, group by rarity*

SELECT

    cc\_rarity AS Rarity,

    GROUP\_CONCAT(cc\_name SEPARATOR '; ') AS CollectionCardNames

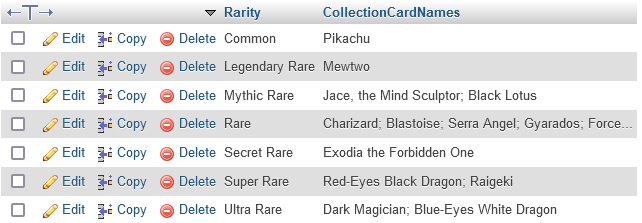
FROM

    CollectionCard

GROUP BY

    cc\_rarity;

Result:



*-- 5 To display employees whose email addresses do not use Gmai*

SELECT \*

FROM employee

WHERE NOT e\_email LIKE '%@gmail.com%';

Result:



*-- Aggregate Query*

*-- 1 Get number of transactions made by each player*

SELECT p.p\_ID AS player\_id, p.p\_name AS player\_name, COUNT(\*) AS transaction\_count

FROM player p

JOIN transaction t ON p.p\_ID = t.t\_p\_ID

GROUP BY p.p\_ID, p.p\_name;

Result:



*-- 2 Get  the total revenue generated from each game room type*

SELECT

    gr.gr\_ID,

    gr.gr\_type,

    SUM(tr.t\_totalPrice) AS total\_revenue,

    COUNT(tr.t\_ID) AS total\_transactions

FROM

    gameroom gr

LEFT JOIN

    roomshoppingcart rsc ON gr.gr\_ID = rsc.rsc\_gr\_ID

LEFT JOIN

    transaction tr ON rsc.rsc\_pt\_ID = tr.t\_ID

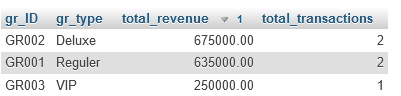
GROUP BY

    gr.gr\_ID, gr.gr\_type

ORDER BY

    total\_revenue DESC;

Result:



*-- 3 List total number of transactions, the average transaction amount, and the player with the highest total, except owner*

SELECT

    player.p\_ID,

    player.p\_name,

    COUNT(transaction.t\_ID) AS total\_transactions,

    AVG(transaction.t\_totalPrice) AS average\_transaction\_amount,

    MAX(transaction.t\_totalPrice) AS highest\_transaction\_amount

FROM

    player

LEFT JOIN

    transaction ON player.p\_ID = transaction.t\_p\_ID

WHERE

    player.p\_ID <> 'OWNER'

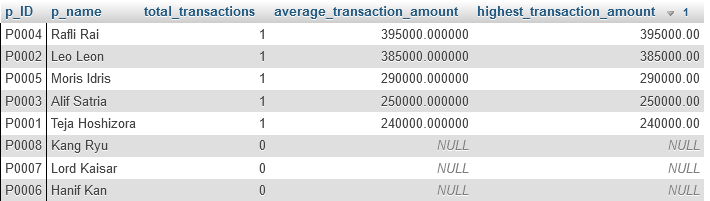
GROUP BY

    player.p\_ID, player.p\_name

ORDER BY

    highest\_transaction\_amount DESC;

Result:



*-- 4 Total pendapatan bulan november*

SELECT SUM(t\_totalPrice) AS totalPendapatan FROM Transaction;

Result:



*-- Nested Query*

*-- 1 Display products that have been sold more than the average quantity sold*

SELECT

    pc.pc\_ID,

    pc.pc\_name,

    pc.pc\_type,

    pc.pc\_price,

    SUM(psc.psc\_quantity) AS total\_sold

FROM

    productcatalog pc

JOIN

    productshoppingcart psc ON pc.pc\_ID = psc.psc\_pc\_ID

JOIN

    transaction t ON psc.psc\_pt\_ID = t.t\_ID

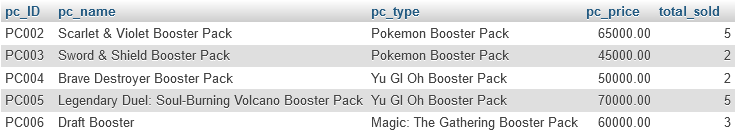
GROUP BY

    pc.pc\_ID

HAVING

    total\_sold > (SELECT AVG(psc\_quantity) FROM productshoppingcart);

Result:



*-- 2 Display the employee name and the total transaction for each employee*

SELECT

    e.e\_name,

    (

        SELECT COUNT(t\_ID)

        FROM Transaction

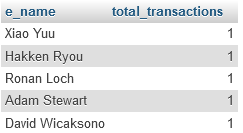
        WHERE t\_e\_ID = e.e\_ID

    ) AS total\_transactions

FROM

    Employee e;

Result:



*-- 3 Search for the player that bought the most expensive product*

SELECT

    t.t\_ID,

    p.p\_name AS player\_name,

    (

        SELECT pc\_name

        FROM ProductCatalog pc

        WHERE pc.pc\_ID = (

            SELECT psc\_pc\_ID

            FROM ProductShoppingCart

            WHERE psc\_pt\_ID = t.t\_ID

            ORDER BY (SELECT pc\_price FROM ProductCatalog WHERE pc\_ID = psc\_pc\_ID) DESC

            LIMIT 1

        )

    ) AS most\_expensive\_product,

    (

        SELECT MAX((SELECT pc\_price FROM ProductCatalog WHERE pc\_ID = psc\_pc\_ID) \* psc\_quantity)

        FROM ProductShoppingCart

        WHERE psc\_pt\_ID = t.t\_ID

    ) AS total\_spent

FROM

    Transaction t

    JOIN Player p ON t.t\_p\_ID = p.p\_ID

ORDER BY total\_spent DESC

LIMIT 1;

Result:



5b. December’s DML

*-- Player Ryu Liam came and brought his friends, then registered for membership.*

INSERT INTO Membership (m\_ID, m\_name, m\_email, m\_dateRegistered, m\_totalPoin) VALUES

('M0008', 'Topazr', 'topaz@yahoo.com', '2023-08-28', 130);

INSERT INTO Player (p\_ID, p\_m\_ID, p\_name, p\_address, p\_phoneNumber) VALUES

('P0009', NULL, 'Ian Harris', 'Jl. Diponegoro No. 606', '555-3456'),

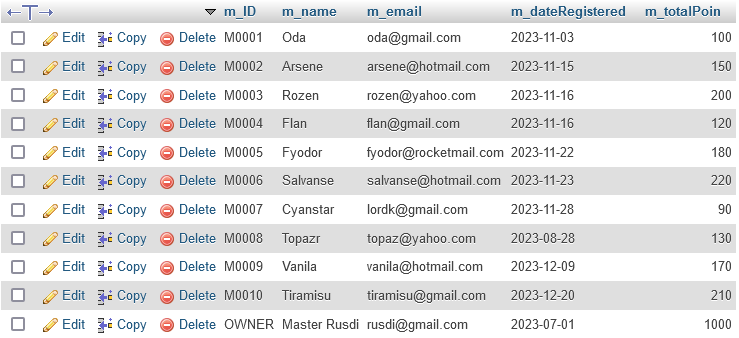
('P0010', NULL, 'Julia Clark', 'Jl. HR. Rasuna Said No. 707', '555-7890');

INSERT INTO Membership (m\_ID, m\_name, m\_email, m\_dateRegistered, m\_totalPoin) VALUES

('M0009', 'Vanila', 'vanila@hotmail.com', '2023-09-09', 170),

('M0010', 'Tiramisu', 'tiramisu@gmail.com', '2023-10-20', 210);

Result:



UPDATE Player

SET p\_m\_ID = 'M0008'

WHERE p\_ID = 'P0008';

UPDATE Player

SET p\_m\_ID = 'M0009'

WHERE p\_ID = 'P0009';

UPDATE Player

SET p\_m\_ID = 'M0010'

WHERE p\_ID = 'P0010';

Result:



*-- New Product: Card Deck Launch*

INSERT INTO ProductCatalog (pc\_ID, pc\_name, pc\_type, pc\_price, pc\_stock) VALUES

('PC008', 'Mew Genesect', 'Pokemon Deck', 110000, 3),

('PC009', 'Inteleon Urshifu', 'Pokemon Deck', 130000, 5);

INSERT INTO ProductCatalog (pc\_ID, pc\_name, pc\_type, pc\_price, pc\_stock) VALUES

('PC010', 'Salamangreat', 'Yu GI Oh Deck', 120000, 1),

('PC011', 'Mannadium Scareclaw Kashtira', 'Yu GI Oh Deck', 135000, 2);

INSERT INTO ProductCatalog (pc\_ID, pc\_name, pc\_type, pc\_price, pc\_stock) VALUES

('PC012', 'Vintage', 'Magic: The Gathering Deck', 100000, 5),

('PC013', 'Alchemy', 'Magic: The Gathering Deck', 110000, 2);

Result:



*-- New Product: Merchandise Launch*

INSERT INTO ProductCatalog (pc\_ID, pc\_name, pc\_type, pc\_price, pc\_stock) VALUES

('PC014', 'Pokemon Hat', 'Type Merchandise', 130000, 3),

('PC015', 'Yu Gi Oh Prism', 'Type Merchandise', 150000, 3),

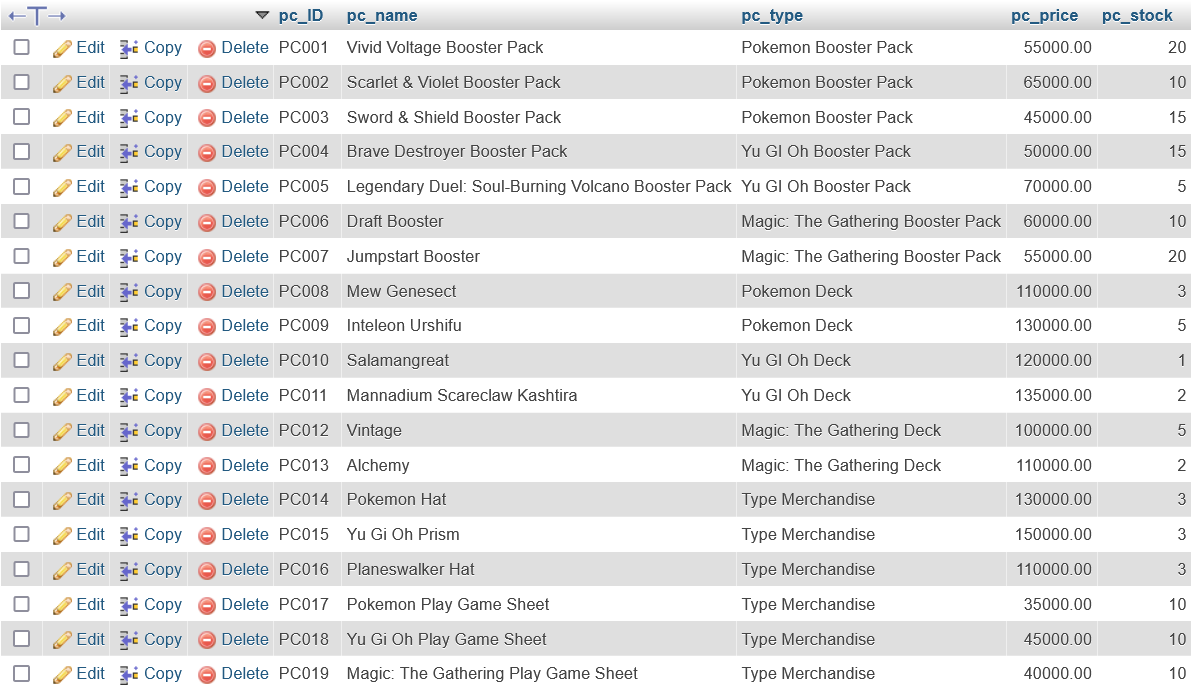
('PC016', 'Planeswalker Hat', 'Type Merchandise', 110000, 3),

('PC017', 'Pokemon Play Game Sheet', 'Type Merchandise', 35000, 10),

('PC018', 'Yu Gi Oh Play Game Sheet', 'Type Merchandise', 45000, 10),

('PC019', 'Magic: The Gathering Play Game Sheet', 'Type Merchandise', 40000, 10);

Result:



*-- Update Stock and replacing Product*

UPDATE ProductCatalog

SET pc\_stock = 10

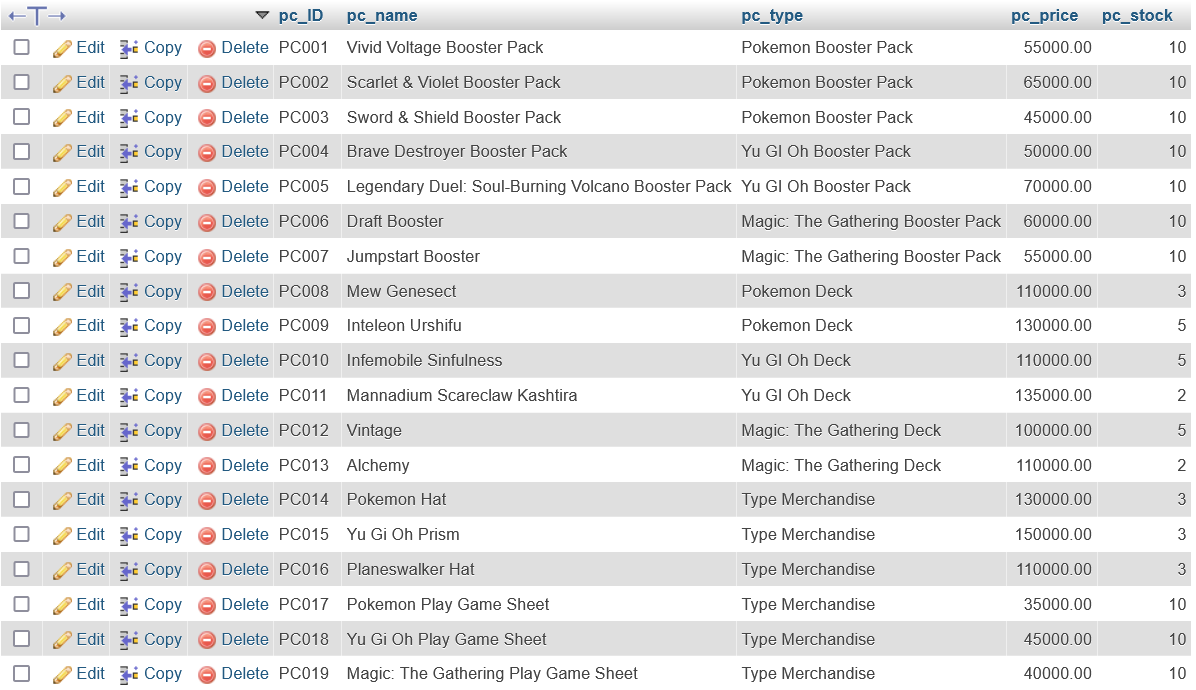
WHERE pc\_type LIKE '%Booster Pack%' AND pc\_name <> 'Salamangreat';

UPDATE ProductCatalog

SET pc\_name = 'Infemobile Sinfulness', pc\_price = 110000, pc\_stock = 5

WHERE pc\_ID = 'PC010';

Result:



*-- transaction in December*

*-- Transaction table*

INSERT INTO Transaction (t\_ID, t\_date, t\_totalPrice, t\_isAMember, t\_e\_ID, t\_p\_ID) VALUES

('T0006', '2023-12-01 10:00:00', 0.00, 1, 'E0001', 'P0008'),

('T0007', '2023-12-02 13:45:00', 0.00, 1, 'E0002', 'P0009'),

('T0008', '2023-12-03 11:15:00', 0.00, 1, 'E0003', 'P0010'),

('T0009', '2023-12-04 14:00:00', 0.00, 1, 'E0004', 'P0001'),

('T0010', '2023-12-05 12:30:00', 0.00, 1, 'E0005', 'P0002');

Result:



*-- ProductShoppingCart table*

    INSERT INTO ProductShoppingCart (psc\_pc\_ID, psc\_pt\_ID, psc\_quantity) VALUES

    ('PC008', 'T0006', 2),

    ('PC014', 'T0006', 1),

    ('PC009', 'T0007', 3),

    ('PC016', 'T0007', 1),

    ('PC011', 'T0008', 2),

    ('PC013', 'T0008', 1),

    ('PC010', 'T0009', 3),

    ('PC015', 'T0009', 2),

    ('PC012', 'T0010', 1),

    ('PC017', 'T0010', 2);

Result:



*-- RoomShoppingCart table*

    INSERT INTO RoomShoppingCart (rsc\_pt\_ID, rsc\_gr\_ID, rsc\_time\_in, rsc\_time\_out) VALUES

    ('T0006', 'GR001', '2023-12-01 10:00:00', '2023-12-01 13:00:00'),

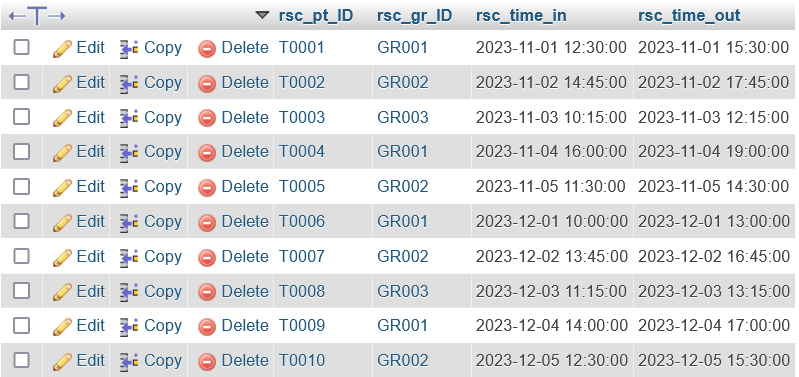
    ('T0007', 'GR002', '2023-12-02 13:45:00', '2023-12-02 16:45:00'),

    ('T0008', 'GR003', '2023-12-03 11:15:00', '2023-12-03 13:15:00'),

    ('T0009', 'GR001', '2023-12-04 14:00:00', '2023-12-04 17:00:00'),

    ('T0010', 'GR002', '2023-12-05 12:30:00', '2023-12-05 15:30:00');

Result:



*-- Update transaction*

    UPDATE Transaction

    SET t\_totalPrice = (

    SELECT SUM(pc\_price \* psc\_quantity)

    FROM ProductShoppingCart

    JOIN ProductCatalog ON ProductShoppingCart.psc\_pc\_ID = ProductCatalog.pc\_ID

    WHERE ProductShoppingCart.psc\_pt\_ID = Transaction.t\_ID

    ) + (

    SELECT SUM(gr\_price \* TIMESTAMPDIFF(SECOND, rsc\_time\_in, rsc\_time\_out) / 3600)

    FROM RoomShoppingCart

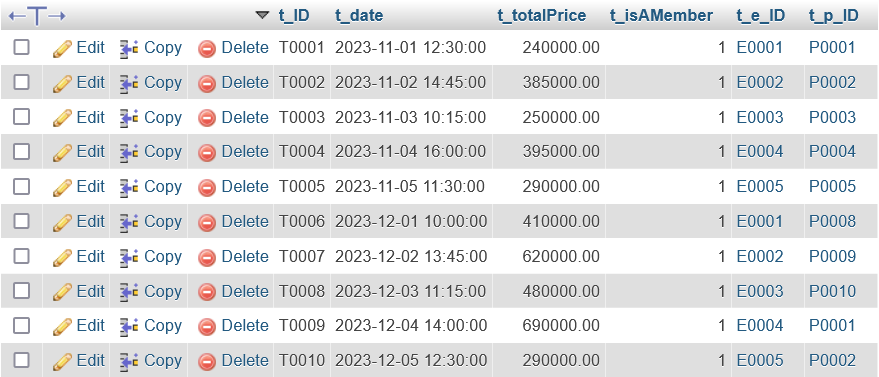
    JOIN GameRoom ON RoomShoppingCart.rsc\_gr\_ID = GameRoom.gr\_ID

    WHERE RoomShoppingCart.rsc\_pt\_ID = Transaction.t\_ID

    )

    WHERE t\_totalPrice = 0;

Result:



6b. December’s Query

*-- Basic Query*

*-- 1 Show approved collection transactions*

SELECT ct.ct\_ID, ct.ct\_date, p1.p\_name AS player1, p2.p\_name AS player2

FROM collectiontransaction ct

JOIN player p1 ON ct.p1\_p\_ID = p1.p\_ID

JOIN player p2 ON ct.p2\_p\_ID = p2.p\_ID

WHERE ct.ct\_isApproved = 1;

Result:



*-- 2 Show the least used game room*

SELECT gr.gr\_ID, gr.gr\_type, COUNT(rsc.rsc\_gr\_ID) AS usage\_count

FROM gameroom gr

JOIN roomshoppingcart rsc ON gr.gr\_ID = rsc.rsc\_gr\_ID

GROUP BY gr.gr\_ID, gr.gr\_type

ORDER BY usage\_count ASC

LIMIT 1;

Result:



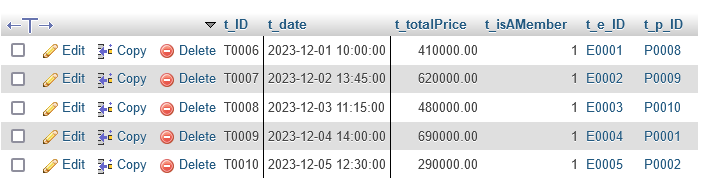
*-- 3 Show all transactions in December*

SELECT \*

FROM transaction

WHERE MONTH(t\_date) = 12;

Result:



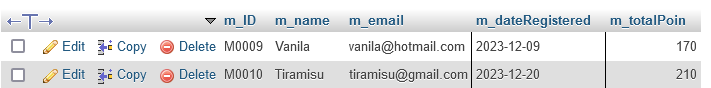
*-- 4 Member created in december*

SELECT \*

FROM membership

WHERE MONTH(m\_dateRegistered) = 12;

Result:



*-- 5 Retrieve employee names and contact information*

SELECT

    e\_name,

    e\_phoneNumber,

    e\_email

FROM

    `employee`

WHERE

    e\_age > 24;

Result:



*-- Aggregate Query*

*-- 1 Find the top 3 players who spent the most money on transactions in December*

SELECT p.p\_name, SUM(t.t\_totalPrice) AS total\_spent

FROM player p

JOIN transaction t ON p.p\_ID = t.t\_p\_ID

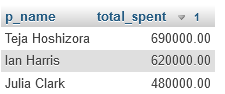
WHERE MONTH(t.t\_date) = 12

GROUP BY p.p\_name

ORDER BY total\_spent DESC

LIMIT 3;

Result:



*-- 2 Show the most used game room by time*

SELECT gr.gr\_ID, gr.gr\_type, SUM(TIMESTAMPDIFF(SECOND, rsc.rsc\_time\_in, rsc.rsc\_time\_out)) AS second\_spent

FROM GameRoom gr

JOIN RoomShoppingCart rsc ON gr.gr\_ID = rsc.rsc\_gr\_ID

GROUP BY gr.gr\_ID, gr.gr\_type

ORDER BY second\_spent DESC

LIMIT 1;

Result:



*-- 3 show the total revenue for each product type*

SELECT

    productcatalog.pc\_type,

    SUM(transaction.t\_totalPrice) AS total\_revenue

FROM

    productcatalog

LEFT JOIN

    productshoppingcart ON productcatalog.pc\_ID = productshoppingcart.psc\_pc\_ID

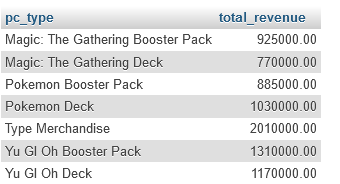
LEFT JOIN

    transaction ON productshoppingcart.psc\_pt\_ID = transaction.t\_ID

GROUP BY

    productcatalog.pc\_type;

Result:



*-- 4 Show employee with the highest number of transaction services*

SELECT Employee.e\_ID, Employee.e\_name, COUNT(Transaction.t\_ID) AS total\_transaction

FROM Employee

JOIN Transaction ON Employee.e\_ID = Transaction.t\_e\_ID

GROUP BY Employee.e\_ID, Employee.e\_name

ORDER BY total\_transaction DESC

LIMIT 1;

Result:



*-- Nested Query*

*-- 1 Retrieve members with a total point above the average*

SELECT \*

FROM membership

WHERE m\_totalPoin > (SELECT AVG(m\_totalPoin) FROM membership);

Result:



*-- 2 Ryu Liam's friend, Ian Harris, wants to pay his transaction here. In December, there is a 10% discount, so his transactions are subject to the discount. Display the total price he needs to pay for all his transa*

SELECT SUM(t\_totalPrice \* 0.9) AS totalTagihan FROM Transaction

WHERE t\_p\_ID = (

    SELECT p\_ID FROM Player

    WHERE p\_name = 'Ian Harris'

);

*ctions.*

Result:



*-- 3 Show the difference in income for November and October*

SELECT

    NovemberIncome,

    DecemberIncome,

    DecemberIncome - NovemberIncome AS IncomeDifference

FROM (

    SELECT

        SUM(t\_totalPrice) AS NovemberIncome

    FROM

        `transaction`

    WHERE

        MONTH(t\_date) = 11

) AS NovemberData,

(

    SELECT

        SUM(t\_totalPrice) AS DecemberIncome

    FROM

        `transaction`

    WHERE

        MONTH(t\_date) = 12

) AS DecemberData;

Result:



Thank You for Reading

&

Have a Good Day Ma’am 😊