week4questions

October 20, 2023

[1]:
```
# Load the ipython-sql extension
%load_ext sql

# Connect to your MySQL database using the mysqlclient driver
%sql mysql://root:12345678@localhost
```

[2]:
```
%sql USE Aetheria
```

```
 * mysql://root:***@localhost
0 rows affected.
```

[2]:
```
[]
```

[ ]:
```
# 1-) Player Behavior Analysis: Identify the five most commonly
#used combinations of first, second, and third items
#in player inventories during quests.
```

[16]:
```
%%sql
SELECT i1.ItemName, i2.ItemName, i3.ItemName
FROM Inventory inv
JOIN Item i1 ON inv.ID = i1.InventoryID
JOIN Item i2 ON inv.ID = i2.InventoryID AND i2.ID > i1.ID
JOIN Item i3 ON inv.ID = i3.InventoryID AND i3.ID > i2.ID
GROUP BY i1.ItemName, i2.ItemName, i3.ItemName
LIMIT 5;
```

```
 * mysql://root:***@localhost
4 rows affected.
```

[16]:
```
[('"Staff of Tranquility"', '"Orb of the Soul Reaver"', ' "Shield" '),
 ('"Armor of the Sea Spirit"', '"Amulet of the Moon"', '"Armor of the Sea
Spirit"'),
 ('"Ring of the Fallen King"', 'Arrow', ' "Sword" '),
 ('"Sunfire Cape"', '"Stormbringer Blade"', ' "Potion" ')]
```

[ ]:
```
# 2-) Chat Metrics: Calculate the average number of messages
# per minute sent in each chat room between 8 PM and 9 PM.
```

[19]:
```
%%sql
SELECT ChatID, COUNT(*) / 60 as AvgMessagesPerMinute
FROM Message
WHERE HOUR(TimeStamp) = 20
GROUP BY ChatID;
```

```
 * mysql://root:***@localhost
3 rows affected.
```

[19]:
```
[(3, Decimal('0.0167')), (30, Decimal('0.0167')), (101, Decimal('0.0167'))]
```

[ ]:
```
# 3-) Rapid Item Acquisition: Identify any player who has acquired
#items with a total value exceeding a certain threshold
#(e.g., 1000 gold) in less than 24 hours.
```

[32]:
```
%%sql
SELECT CharacterID
FROM (
    SELECT
        CharacterID,
        DATE(Time) as TransactionDate,
        SUM(Amount) as TotalValue
    FROM Transaction
    GROUP BY CharacterID, TransactionDate
) AS DailyTotal
WHERE TotalValue > 1000;
```

```
 * mysql://root:***@localhost
46 rows affected.
```

[32]:
```
[(146,),
 (98,),
 (148,),
 (127,),
 (23,),
 (230,),
 (49,),
 (133,),
 (49,),
 (32,),
 (200,),
 (49,),
 (99,),
 (41,),
 (233,),
 (119,),
 (206,),
```

```
(137,),
(140,),
(76,),
(202,),
(166,),
(180,),
(18,),
(212,),
(160,),
(83,),
(206,),
(53,),
(104,),
(163,),
(191,),
(137,),
(175,),
(10,),
(133,),
(140,),
(18,),
(109,),
(98,),
(227,),
(96,),
(84,),
(251,),
(134,),
(32,)]
```

[ ]:
```
#4-) Large Money Transfers: Highlight transactions where large sums of in-game
#currency are transferred between accounts that are not in the same guild or
#have not interacted before.

#This query considers the relationship between the
#sender (c1) and the item owner
#(OwnerID from the Inventory table) and checks if their guilds
#are different. It should now correctly highlight transactions where
#large sums are transferred between characters with different guilds.
```

[50]:
```
%%sql
SELECT c1.ID as SenderID, t.CharacterID as ReceiverID, t.Amount
FROM Transaction t
JOIN `Character` c1 ON c1.ID = (SELECT OwnerID FROM Inventory WHERE ID = t.
 ↪ItemID)
WHERE t.Amount > 1700
AND NOT EXISTS (
```

```
    SELECT 1
    FROM `Character` c2
    WHERE c1.GuildID = c2.GuildID AND c2.ID = t.CharacterID
);
```

```
 * mysql://root:***@localhost
11 rows affected.
```

[50]:
```
[(167, 146, 1980),
 (179, 148, 1740),
 (167, 49, 1860),
 (127, 49, 1960),
 (61, 140, 1960),
 (127, 18, 1760),
 (204, 83, 1940),
 (149, 137, 1780),
 (52, 10, 1860),
 (160, 133, 1800),
 (146, 96, 1880)]
```

[51]:
```
# 5-) Top Traded Items: Identify which items are most commonly traded or sold␣
 ↪in-game.
```

[54]:
```
%%sql
SELECT i.ItemName, COUNT(*) as TradeFrequency
FROM Transaction t
JOIN Item i ON t.ItemID = i.ID
GROUP BY i.ItemName
ORDER BY TradeFrequency DESC
LIMIT 10;
```

```
 * mysql://root:***@localhost
10 rows affected.
```

[54]:
```
[('Arrow', 12),
 (' "Sword" ', 7),
 (' "Potion" ', 6),
 ('"Orb of the Soul Reaver"', 5),
 (' "Bow" ', 5),
 ('"Stormbringer Blade"', 5),
 ('"Dragon\'s Breath Bow"', 4),
 (' "Shield" ', 3),
 ('"Rod of Wondrous Echoes"', 3),
 ('"Warrior\'s Battleaxe"', 3)]
```

[55]:
```
# 6-) Market Hotspots: Find locations in the game where the most transactions␣
 ↪or trades occur.
```

[76]:
```
%%sql
SELECT
    CONCAT(
        FLOOR(Location / 1),
        ' - ',
        (FLOOR(Location / 1) +3)
    ) AS LocationInterval,
    COUNT(*) AS TransactionCount
FROM
    Transaction
GROUP BY
    LocationInterval
ORDER BY
    TransactionCount DESC
LIMIT 5;
```

```
 * mysql://root:***@localhost
5 rows affected.
```

[76]:
```
[('8 - 11', 11), ('13 - 16', 9), ('11 - 14', 7), ('6 - 9', 7), ('3 - 6', 7)]
```

[77]:
```
# 7-) Rare Item Ownership: List players who own the rarest items in the game.
```

[85]:
```
%%sql
SELECT i.ID, i.ItemName
FROM `Character` c
JOIN Inventory inv ON c.ID = inv.OwnerID
JOIN Item i ON inv.ID = i.InventoryID
WHERE i.ID IN (
    SELECT ItemID
    FROM (
        SELECT ItemID, ROW_NUMBER() OVER (ORDER BY COUNT(*) ASC) as row_num
        FROM Transaction
        GROUP BY ItemID
    ) ranked
    WHERE row_num <= 5
);
```

```
 * mysql://root:***@localhost
5 rows affected.
```

[85]:
```
[(4, '"Quiver of the Wind"'),
 (307, ' "Potion" '),
 (9, '"Cloak of Elvenkind"'),
 (305, ' "Bow" '),
 (15, '"Crystal of the Ancients"')]
```

[86]:
```
# 8-) Frequent Guild Changes: Flag players who have joined and left more than␣
↪'n' guilds within a 7-day period.
```

[94]:
```
%%sql
SELECT CharacterID, COUNT(DISTINCT GuildID) AS NumGuildChanges
FROM GuildMembershipLog
WHERE LeaveDate IS NOT NULL
GROUP BY CharacterID
HAVING COUNT(DISTINCT GuildID) > 0
   AND MAX(JoinDate) >= NOW() - INTERVAL 7 DAY;
```

```
 * mysql://root:***@localhost
2 rows affected.
```

[94]:
```
[(65, 1), (122, 1)]
```

[ ]:
```
# 9-) High Frequency Quest Completions:
# Detect players who have completed more than 'x' quests within a 24-hour␣
↪period.
```

[99]:
```
%%sql
SELECT
    CharacterID,
    COUNT(*) AS NumQuestCompletions
FROM
    QuestLog
WHERE
    CompletionTime >= NOW() - INTERVAL 24 HOUR
GROUP BY
    CharacterID
HAVING
    COUNT(*) > 2;
```

```
 * mysql://root:***@localhost
3 rows affected.
```

[99]:
```
[(5, 3), (160, 3), (203, 3)]
```

[100]:
```
# 10-) Trending Quests: List the top 3 quests that have shown the
#most significant percentage increase in player participation in the last month.
```

[103]:
```
%%sql
WITH QuestParticipation AS (
    SELECT
        q.ID AS QuestID,
        COUNT(DISTINCT ql.CharacterID) AS NumParticipantsLastMonth
    FROM
```

```
        Quest q
    LEFT JOIN
        QuestLog ql ON q.ID = ql.QuestID AND ql.CompletionTime >= NOW() -␣
 ↪INTERVAL 1 MONTH
    GROUP BY
        q.ID
)

SELECT
    q.ID AS QuestID,
    q.Description,
    q.RegionID,
    q.Type,
    q.ItemID,
    qp.NumParticipantsLastMonth
FROM
    Quest q
JOIN
    QuestParticipation qp ON q.ID = qp.QuestID
ORDER BY
    NumParticipantsLastMonth DESC
LIMIT 3;
```

```
 * mysql://root:***@localhost
3 rows affected.
```

[103]:
```
[(91, 'Quest 9', 1, 'Exploration', 25, 3),
 (4, 'Quest 6', 1, 'Gathering', 34, 3),
 (5, 'Quest 5', 3, 'Gathering', 39, 3)]
```

[104]:
```
# 11-) High Dropout Rates: Find quests with a high start but low completion␣
 ↪rate.
```

[111]:
```
%%sql
SELECT
    q.ID AS QuestID,
    q.Description,
    q.RegionID,
    q.Type,
    q.ItemID  -- Assuming ItemID is the correct column for the item
FROM
    Quest q
LEFT JOIN (
    SELECT
        QuestID,
        COUNT(DISTINCT CASE WHEN CompletionTime IS NOT NULL THEN CharacterID␣
 ↪END) AS NumCompleted,
```

```
        COUNT(DISTINCT CharacterID) AS NumStarted,
        COALESCE(COUNT(DISTINCT CASE WHEN CompletionTime IS NOT NULL THEN␣
 ↪CharacterID END) /
                 NULLIF(COUNT(DISTINCT CharacterID), 0), 0) AS CompletionRate
    FROM
        QuestLog
    GROUP BY
        QuestID
) qcr ON q.ID = qcr.QuestID
ORDER BY
    qcr.CompletionRate ASC, qcr.NumStarted DESC
LIMIT 5;
```

```
 * mysql://root:***@localhost
5 rows affected.
```

[111]:
```
[(33, 'Quest 7', 3, 'Gathering', 18),
 (99, 'Quest 1', 3, 'Gathering', 32),
 (3, 'Quest 7', 1, 'Exploration', 44),
 (44, 'Quest 6', 3, 'Gathering', 9),
 (43, 'Quest 7', 2, 'Gathering', 22)]
```

[112]:
```
# 12-) Time of Play: Find out what times of day are the most popular for␣
 ↪playing.
```

[122]:
```
%%sql
SELECT
    HOUR(LastLogin) AS HourOfDay,
    COUNT(*) AS NumPlayers
FROM
    `Character`
GROUP BY
    HourOfDay
ORDER BY
    NumPlayers DESC
LIMIT 5;
```

```
 * mysql://root:***@localhost
5 rows affected.
```

[122]:
```
[(21, 11), (3, 10), (7, 10), (15, 9), (2, 9)]
```

[123]:
```
# 13-) Level Distribution: Examine the distribution of player levels.
```

[135]:
```
%%sql
SELECT Level, COUNT(*) AS NumCharacters
FROM `Character`
```

```
GROUP BY Level
ORDER BY Level;
```

```
 * mysql://root:***@localhost
78 rows affected.
```

[135]:
```
[(0, 1),
 (1, 1),
 (3, 2),
 (4, 3),
 (5, 2),
 (6, 2),
 (8, 2),
 (9, 1),
 (11, 2),
 (12, 3),
 (13, 1),
 (14, 2),
 (17, 2),
 (18, 1),
 (19, 1),
 (20, 1),
 (21, 3),
 (23, 1),
 (25, 3),
 (26, 1),
 (27, 3),
 (28, 1),
 (29, 2),
 (30, 1),
 (32, 1),
 (34, 2),
 (35, 1),
 (36, 1),
 (37, 2),
 (38, 2),
 (39, 1),
 (40, 2),
 (41, 1),
 (42, 1),
 (43, 3),
 (44, 1),
 (45, 3),
 (46, 1),
 (47, 2),
 (49, 1),
 (51, 2),
```

```
 (54, 2),
 (55, 1),
 (56, 2),
 (57, 6),
 (58, 1),
 (59, 2),
 (60, 4),
 (63, 2),
 (64, 5),
 (65, 1),
 (66, 1),
 (67, 3),
 (68, 3),
 (69, 1),
 (70, 1),
 (71, 4),
 (73, 1),
 (76, 1),
 (77, 2),
 (78, 2),
 (79, 1),
 (80, 1),
 (81, 2),
 (83, 4),
 (84, 2),
 (85, 1),
 (86, 1),
 (87, 1),
 (89, 2),
 (90, 1),
 (92, 2),
 (93, 1),
 (94, 1),
 (95, 3),
 (97, 2),
 (98, 2),
 (100, 1)]
```

[136]:
```
## 14-) Player Retention: Analyze the average session length and frequency for␣
 ↪each player.
```

[151]:
```
%%sql
SELECT
    ID,
    AVG(TIMESTAMPDIFF(MINUTE, LastLogin, EnterOut)) DIV 60000.5 as␣
 ↪AvgSessionLengthHours
FROM `Character`
```

```
GROUP BY ID
HAVING AvgSessionLengthHours > 0;
```

```
 * mysql://root:***@localhost
61 rows affected.
```

[151]:
```
[(2, 4),
 (6, 1),
 (10, 1),
 (14, 3),
 (15, 2),
 (18, 1),
 (21, 4),
 (22, 1),
 (29, 5),
 (32, 1),
 (35, 2),
 (49, 1),
 (52, 3),
 (54, 5),
 (56, 6),
 (61, 3),
 (71, 2),
 (74, 2),
 (76, 3),
 (78, 3),
 (84, 2),
 (86, 2),
 (90, 3),
 (92, 5),
 (95, 7),
 (100, 6),
 (104, 4),
 (107, 5),
 (109, 3),
 (116, 4),
 (119, 1),
 (127, 3),
 (129, 4),
 (133, 3),
 (134, 5),
 (137, 1),
 (139, 7),
 (140, 2),
 (145, 1),
 (148, 4),
 (152, 4),
```

```
 (160, 3),
 (172, 3),
 (173, 1),
 (178, 4),
 (180, 6),
 (186, 2),
 (188, 1),
 (189, 1),
 (193, 3),
 (194, 4),
 (202, 4),
 (209, 1),
 (226, 6),
 (230, 1),
 (238, 6),
 (241, 1),
 (249, 2),
 (251, 7),
 (255, 2),
 (260, 2)]
```

[152]:
```
## 15-) Inflation Analysis: Track the average value of traded items over time␣
 ↪to detect if the in-game economy is experiencing inflation or deflation.
```

[160]:
```
%%sql
SELECT
    Month,
    AVG(AverageTradeValue) AS AverageTradeValue,
    LAG(AVG(AverageTradeValue)) OVER (ORDER BY Month) AS␣
 ↪PreviousMonthAvgTradeValue,
    AVG(AverageTradeValue) - LAG(AVG(AverageTradeValue)) OVER (ORDER BY Month)␣
 ↪AS MonthlyDifference
FROM (
    SELECT
        DATE_FORMAT(t.Time, '%Y-%m') AS Month,
        AVG(t.Amount * i.Value/10000) AS AverageTradeValue
    FROM
        Transaction t
    JOIN
        Item i ON t.ItemID = i.ID
    GROUP BY
        Month
) AS MonthlyTradeData
GROUP BY
    Month
ORDER BY
    Month;
```

```
 * mysql://root:***@localhost
13 rows affected.
```

[160]:
```
[('2022-10', Decimal('52.679500000000'), None, None),
 ('2022-11', Decimal('68.816333330000'), Decimal('52.679500000000'),
Decimal('16.136833330000')),
 ('2022-12', Decimal('34.378000000000'), Decimal('68.816333330000'),
Decimal('-34.438333330000')),
 ('2023-01', Decimal('78.581111110000'), Decimal('34.378000000000'),
Decimal('44.203111110000')),
 ('2023-02', Decimal('55.410444440000'), Decimal('78.581111110000'),
Decimal('-23.170666670000')),
 ('2023-03', Decimal('56.369571430000'), Decimal('55.410444440000'),
Decimal('0.959126990000')),
 ('2023-04', Decimal('31.914750000000'), Decimal('56.369571430000'),
Decimal('-24.454821430000')),
 ('2023-05', Decimal('51.481428570000'), Decimal('31.914750000000'),
Decimal('19.566678570000')),
 ('2023-06', Decimal('35.339333330000'), Decimal('51.481428570000'),
Decimal('-16.142095240000')),
 ('2023-07', Decimal('50.942181820000'), Decimal('35.339333330000'),
Decimal('15.602848490000')),
 ('2023-08', Decimal('63.357555560000'), Decimal('50.942181820000'),
Decimal('12.415373740000')),
 ('2023-09', Decimal('34.156500000000'), Decimal('63.357555560000'),
Decimal('-29.201055560000')),
 ('2023-10', Decimal('31.394666670000'), Decimal('34.156500000000'),
Decimal('-2.761833330000'))]
```

[ ]: