

# 🚖 Problem 1: Uber – Identify Consistent Riders in Chennai

## 🧩 Question (Tanglish)

Gowtham wants to find Uber customers (like Mani, Ravi) who completed at least 50 rides in Chennai over the last 3 months without any cancellations.

## 🗂 Problem Statement (Tanglish)

Uber-ku customer loyalty track panna, yaar consistent-ah Chennai city-la ride edukkaanga, andha customers find pannanum. Condition: last 3 months-la at least 50 rides complete panni irukkanum, and cancellation irukka koodaadhu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE uber\_rides ( ride\_id INT PRIMARY KEY, customer\_id INT, customer\_name VARCHAR(50), city VARCHAR(50),

ride\_date DATE, cancelled BOOLEAN

);

INSERT INTO uber\_rides VALUES

(1, 201, 'Mani', 'Chennai', '2025-04-01', FALSE),

(2, 201, 'Mani', 'Chennai', '2025-04-02', FALSE),

-- (add 48 more rows for Mani)

(51, 202, 'Ravi', 'Chennai', '2025-04-01', FALSE),

(52, 202, 'Ravi', 'Chennai', '2025-04-02', TRUE),

-- (add additional data for variety)

(100, 203, 'Arun', 'Coimbatore', '2025-04-01', FALSE);

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS total\_rides FROM uber\_rides

WHERE city = 'Chennai' AND cancelled = FALSE

AND ride\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

GROUP BY customer\_id, customer\_name HAVING COUNT(\*) >= 50;

## 🩻 Explanation (Tanglish)

### ✅ Step-by-Step:



1

WHERE city = 'Chennai' – Chennai rides maathiri filter pannrom.

cancelled = FALSE – Cancelled rides exclude pannrom. 3 ride\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH) – Last 3 months data maathiri filter. 4 GROUP BY customer\_id, customer\_name – Customer-wise group pannrom. 5 HAVING COUNT(\*) >= 50 – At least 50 completed rides irukka nu check pannrom.



2

### Eg:

* Mani completed 50 rides without cancellations ✅ will be shown.
* Ravi had 50 rides but 1 cancellation ❌ will not be shown.

## 🚖 Use Case Value (Tanglish)

Uber and Gowtham can use this query to identify **consistent riders** for rewards, loyalty programs, and churn prevention for **premium Chennai customers**.

# 🍔 Problem 2: Zomato – Identify Friday Frequent Customers

## 🧩 Question (Tanglish)

Gowtham wants to find Zomato customers (like Karthik, Suresh) who placed orders on every Friday for the last 3 months continuously.

## 🗂 Problem Statement (Tanglish)

Zomato-la customers Friday special offers-ku consistent-ah order panraangala-nu check pannanum. Condition: last 3 months-la, every Friday atleast 1 order panni irukkanum.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE zomato\_orders ( order\_id INT PRIMARY KEY, customer\_id INT, customer\_name VARCHAR(50), order\_date DATE,

amount DECIMAL(10,2)

);

INSERT INTO zomato\_orders VALUES (1, 301, 'Karthik', '2025-04-04', 350.00),

(2, 301, 'Karthik', '2025-04-11', 420.00),

(3, 301, 'Karthik', '2025-04-18', 250.00),

(4, 301, 'Karthik', '2025-04-25', 500.00),

-- (add Fridays for May and June for Karthik) (20, 302, 'Suresh', '2025-04-04', 300.00),

(21, 302, 'Suresh', '2025-04-18', 450.00),

-- (incomplete Fridays for Suresh)

(40, 303, 'Divya', '2025-04-05', 600.00);

## 💡 Solution (MySQL)

WITH fridays AS (

SELECT DISTINCT DATE(order\_date) AS order\_date FROM zomato\_orders

WHERE DAYOFWEEK(order\_date) = 6

AND order\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

),

customer\_fridays AS (

SELECT customer\_id, customer\_name, COUNT(DISTINCT DATE(order\_date)) AS friday\_orders

FROM zomato\_orders

WHERE DAYOFWEEK(order\_date) = 6

AND order\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

GROUP BY customer\_id, customer\_name

),

total\_fridays AS (

SELECT COUNT(\*) AS total\_fridays FROM fridays

)

SELECT c.customer\_id, c.customer\_name FROM customer\_fridays c, total\_fridays t WHERE c.friday\_orders = t.total\_fridays;

## 🩻 Explanation (Tanglish)

### ✅ Step-by-Step:

1 fridays CTE – Last 3 months-la Friday dates collect pannrom.

customer\_fridays CTE – Each customer ethana Fridays orders pannanga-nu count



2

pannrom.

3 total\_fridays CTE – Total Fridays count pannrom (for comparison).

4 Final SELECT-la, c.friday\_orders = t.total\_fridays vechutu, customers every Friday order pannirukka nu filter pannrom.

### Eg:

* Karthik every Friday order pannirukkiraar ✅ will be shown.
* Suresh some Fridays skip pannirukkiraar ❌ will not be shown.

## 🍔 Use Case Value (Tanglish)

Zomato and Gowtham can use this to identify **Friday consistent customers** for special offers, loyalty rewards, and targeted marketing during **weekend peak orders**.

# 📦 Problem 3: Amazon – Top Repeat Purchasers

## 🧩 Question (Tanglish)

Gowtham wants to find Amazon customers (like Priya, Arjun) who purchased the same product at least 5 times in the last 3 months.

## 🗂 Problem Statement (Tanglish)

Amazon-la customers ore product 5 times illa adhu vida adhigama vaangirukkaangala-nu identify panna vendiyathu for last 3 months.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_orders ( order\_id INT PRIMARY KEY, customer\_id INT, customer\_name VARCHAR(50), product\_id INT,

product\_name VARCHAR(50), order\_date DATE

);

INSERT INTO amazon\_orders VALUES

(1, 401, 'Priya', 1001, 'Bluetooth Speaker', '2025-04-02'),

(2, 401, 'Priya', 1001, 'Bluetooth Speaker', '2025-04-10'),

(3, 401, 'Priya', 1001, 'Bluetooth Speaker', '2025-04-20'),

(4, 401, 'Priya', 1001, 'Bluetooth Speaker', '2025-05-01'),

(5, 401, 'Priya', 1001, 'Bluetooth Speaker', '2025-05-15'),

(6, 402, 'Arjun', 1002, 'Wireless Mouse', '2025-04-05'),

(7, 402, 'Arjun', 1002, 'Wireless Mouse', '2025-06-01');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, product\_id, product\_name, COUNT(\*) AS purchase\_count

FROM amazon\_orders

WHERE order\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

GROUP BY customer\_id, customer\_name, product\_id, product\_name HAVING purchase\_count >= 5;

## 🩻 Explanation (Tanglish)

### ✅ Step-by-Step:

1. WHERE order\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH) vechutu last 3 months data filter pannrom.
2. GROUP BY customer\_id, customer\_name, product\_id, product\_name vechutu group panna rom.
3. HAVING purchase\_count >= 5 vechutu same product 5 times illa adhigama vaangina customers filter pannrom.

### Eg:

* + Priya Bluetooth Speaker 5 times vaangirukkanga ✅ will be shown.
  + Arjun Wireless Mouse only 2 times vaangirukkanga ❌ will not be shown.

## 📦 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **repeat product purchasers** to give personalized offers, product bundles, or membership rewards for improving customer loyalty.

# 🎬 Problem 4: Netflix – Identify Binge Watchers

## 🧩 Question (Tanglish)

Gowtham wants to find Netflix users (like Saravanan, Divya) who watched at least 5 different movies per week for 4 consecutive weeks.

## 🗂 Problem Statement (Tanglish)

Netflix-la binge watchers identify panna vendiyathu, last 4 weeks continuous-ah, weekly 5 different movies watch pannirukkangala-nu check panna.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE netflix\_watch\_history ( watch\_id INT PRIMARY KEY, user\_id INT,

user\_name VARCHAR(50), movie\_id INT,

movie\_name VARCHAR(100), watch\_date DATE

);

INSERT INTO netflix\_watch\_history VALUES

(1, 501, 'Saravanan', 2001, 'Movie A', '2025-06-02'),

(2, 501, 'Saravanan', 2002, 'Movie B', '2025-06-03'),

(3, 501, 'Saravanan', 2003, 'Movie C', '2025-06-04'),

(4, 501, 'Saravanan', 2004, 'Movie D', '2025-06-05'),

(5, 501, 'Saravanan', 2005, 'Movie E', '2025-06-06'),

(6, 502, 'Divya', 2001, 'Movie A', '2025-06-02'),

(7, 502, 'Divya', 2002, 'Movie B', '2025-06-02');

## 💡 Solution (MySQL)

WITH week\_data AS ( SELECT

user\_id, user\_name,

WEEK(watch\_date) AS watch\_week, COUNT(DISTINCT movie\_id) AS movie\_count

FROM netflix\_watch\_history

WHERE watch\_date >= DATE\_SUB(CURDATE(), INTERVAL 4 WEEK)

GROUP BY user\_id, user\_name, WEEK(watch\_date)

)

SELECT user\_id, user\_name

FROM week\_data WHERE movie\_count >= 5

GROUP BY user\_id, user\_name HAVING COUNT(\*) = 4;

## 🩻 Explanation (Tanglish)

### ✅ Step-by-Step:

1. WEEK(watch\_date) use panni weekly partition pannrom.
2. Last 4 weeks data maathiri filter pannrom.
3. Each week 5 different movies watch pannangala-nu movie\_count >= 5 vechutu filter pannrom.
4. HAVING COUNT(\*) = 4 vechutu 4 consecutive weeks satisfy pannra users identify pannrom.

### Eg:

* + Saravanan 5 movies per week for 4 weeks watch pannirundhaal ✅ will be shown.
  + Divya only 2 movies watch pannirukkanga ❌ will not be shown.

## 🎬 Use Case Value (Tanglish)

Netflix and Gowtham can use this to identify **heavy binge watchers** for premium upsell, early access promotions, and loyalty rewards.

# 🍔 Problem 5: Swiggy – Identify Frequent Order Cancellers

## 🧩 Question (Tanglish)

Gowtham wants to find Swiggy customers (like Meena, Ravi) who cancelled orders more than 3 times in any single month.

## 🗂 Problem Statement (Tanglish)

Swiggy-ku frequent cancellers identify panna vendiyathu. Last 6 months-la yaarum oru month-ku mela 3 orders cancel pannirukkangala-nu check panna.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE swiggy\_orders ( order\_id INT PRIMARY KEY, customer\_id INT, customer\_name VARCHAR(50), order\_date DATE,

cancelled BOOLEAN

);

INSERT INTO swiggy\_orders VALUES (1, 601, 'Meena', '2025-03-02', TRUE),

(2, 601, 'Meena', '2025-03-05', TRUE),

(3, 601, 'Meena', '2025-03-10', TRUE),

(4, 601, 'Meena', '2025-03-15', TRUE),

(5, 602, 'Ravi', '2025-04-03', TRUE),

(6, 602, 'Ravi', '2025-04-10', FALSE),

(7, 602, 'Ravi', '2025-04-18', TRUE);

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, DATE\_FORMAT(order\_date, '%Y-%m') AS cancel\_month, COUNT(\*) AS cancel\_count

FROM swiggy\_orders WHERE cancelled = TRUE

GROUP BY customer\_id, customer\_name, DATE\_FORMAT(order\_date, '%Y-%m') HAVING cancel\_count > 3;

## 🩻 Explanation (Tanglish)

### ✅ Step-by-Step:

1. WHERE cancelled = TRUE vechutu cancelled orders mattum filter pannrom.
2. DATE\_FORMAT(order\_date, '%Y-%m') vechutu month-wise group panna ready pannrom.
3. GROUP BY customer\_id, customer\_name, month panna rom.
4. HAVING cancel\_count > 3 vechutu andha month-la 3 mela cancel pannina customers identify pannrom.

### Eg:

* + Meena March month-la 4 orders cancel pannirukkanga ✅ will be shown.
  + Ravi maximum 2 cancel pannirukkanga ❌ will not be shown.

## 🍔 Use Case Value (Tanglish)

Swiggy and Gowtham can use this to identify **frequent cancellers** and take corrective actions like order confirmation calls or delivery prioritisation for improving operational efficiency.

# 📱 Problem 6: Meta – Consistent Post Engagement

## 🧩 Question (Tanglish)

Gowtham wants to find Meta users (like Kavin, Divya) who liked at least 50 posts each month consistently for the last 4 months.

## 🗂 Problem Statement (Tanglish)

Meta-la yaar consistent-ah monthly 50 post-ku mela like pannraangalo, last 4 months continuous-ah, andha users identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE meta\_likes ( like\_id INT PRIMARY KEY, user\_id INT,

user\_name VARCHAR(50), post\_id INT,

like\_date DATE

);

INSERT INTO meta\_likes VALUES (1, 701, 'Kavin', 9001, '2025-03-05'),

(2, 701, 'Kavin', 9002, '2025-03-06'),

-- (add 48 more likes in March for Kavin) (51, 701, 'Kavin', 9051, '2025-04-02'),

-- (add 50 likes in April, May, June for Kavin) (200, 702, 'Divya', 9101, '2025-03-10'),

(201, 702, 'Divya', 9102, '2025-03-12');

## 💡 Solution (MySQL)

WITH monthly\_likes AS (

SELECT user\_id, user\_name, DATE\_FORMAT(like\_date, '%Y-%m') AS like\_month, COUNT(\*) AS like\_count

FROM meta\_likes

WHERE like\_date >= DATE\_SUB(CURDATE(), INTERVAL 4 MONTH)

GROUP BY user\_id, user\_name, DATE\_FORMAT(like\_date, '%Y-%m')

)

SELECT user\_id, user\_name FROM monthly\_likes WHERE like\_count >= 50

GROUP BY user\_id, user\_name HAVING COUNT(\*) = 4;

## 🩻 Explanation (Tanglish)

### ✅ Step-by-Step:

1. WHERE like\_date >= DATE\_SUB(CURDATE(), INTERVAL 4 MONTH) vechutu last 4 months data filter pannrom.
2. DATE\_FORMAT(like\_date, '%Y-%m') vechutu month-wise group pannrom.
3. COUNT(\*) AS like\_count to count likes per month.
4. WHERE like\_count >= 50 filter panni 50 likes cross pannina months maathiri retain pannrom.
5. HAVING COUNT(\*) = 4 vechutu 4 months continuous satisfy pannra users identify pannrom.

### Eg:

* + Kavin 50+ likes for 4 months continuously pannirukkanga ✅ will be shown.
  + Divya only few likes pannirukkanga ❌ will not be shown.

## 📱 Use Case Value (Tanglish)

Meta and Gowtham can use this to identify **highly engaged users** for reel boosts, exclusive badges, and personalized engagement campaigns.

# 🚖 Problem 7: Uber – Identify Early Morning Riders

## 🧩 Question (Tanglish)

Gowtham wants to find Uber customers (like Prabhu, Siva) who took rides between 6-9 AM continuously for 30 days.

## 🗂 Problem Statement (Tanglish)

Uber-la 6-9 AM la daily ride edukka consistent customers identify panna vendiyathu for 30 consecutive days.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE uber\_rides\_morning ( ride\_id INT PRIMARY KEY, customer\_id INT,

customer\_name VARCHAR(50), ride\_date DATE,

ride\_time TIME

);

INSERT INTO uber\_rides\_morning VALUES (1, 801, 'Prabhu', '2025-06-01', '06:30:00'),

(2, 801, 'Prabhu', '2025-06-02', '07:15:00'),

-- (add 28 more for Prabhu)

(31, 802, 'Siva', '2025-06-01', '08:45:00');

## 💡 Solution (MySQL)

WITH morning\_rides AS (

SELECT customer\_id, customer\_name, ride\_date FROM uber\_rides\_morning

WHERE HOUR(ride\_time) BETWEEN 6 AND 8

GROUP BY customer\_id, customer\_name, ride\_date

),

ride\_counts AS (

SELECT customer\_id, customer\_name, COUNT(\*) AS ride\_days FROM morning\_rides

GROUP BY customer\_id, customer\_name

)

SELECT customer\_id, customer\_name FROM ride\_counts

WHERE ride\_days = 30;

## 🩻 Explanation (Tanglish)

### ✅ Step-by-Step:

1. HOUR(ride\_time) BETWEEN 6 AND 8 vechutu 6-9 AM rides filter pannrom.
2. GROUP BY customer\_id, customer\_name, ride\_date vechutu unique days capture pannrom.
3. COUNT(\*) AS ride\_days panna total days count pannrom.
4. WHERE ride\_days = 30 vechutu 30 consecutive days la ride edukka consistent customers identify pannrom.

### Eg:

* + Prabhu 30 days la continuous ride pannirukkanga ✅ will be shown.
  + Siva single day pannirukkanga ❌ will not be shown.

## 🚖 Use Case Value (Tanglish)

Uber and Gowtham can use this to identify **consistent early morning commuters** for special pass offers and premium user tagging.

# 🍔 Problem 8: Swiggy – Identify Late Night Foodies

## 🧩 Question (Tanglish)

Gowtham wants to find Swiggy customers (like Aravind, Selvi) who ordered food after 11 PM for 10 consecutive days.

## 🗂 Problem Statement (Tanglish)

Swiggy-la late night la 11 PM ku mela order pannra customers identify panna vendiyathu for 10 days continuous.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE swiggy\_late\_orders ( order\_id INT PRIMARY KEY, customer\_id INT,

customer\_name VARCHAR(50),

order\_date DATE, order\_time TIME

);

INSERT INTO swiggy\_late\_orders VALUES (1, 901, 'Aravind', '2025-06-01', '23:15:00'),

(2, 901, 'Aravind', '2025-06-02', '23:45:00'),

-- (add 8 more for Aravind)

(11, 902, 'Selvi', '2025-06-01', '22:55:00');

## 💡 Solution (MySQL)

WITH late\_orders AS (

SELECT customer\_id, customer\_name, order\_date FROM swiggy\_late\_orders

WHERE HOUR(order\_time) >= 23

GROUP BY customer\_id, customer\_name, order\_date

),

order\_counts AS (

SELECT customer\_id, customer\_name, COUNT(\*) AS late\_days FROM late\_orders

GROUP BY customer\_id, customer\_name

)

SELECT customer\_id, customer\_name FROM order\_counts

WHERE late\_days = 10;

## 🩻 Explanation (Tanglish)

### ✅ Step-by-Step:

1. HOUR(order\_time) >= 23 vechutu 11 PM ku mela orders filter pannrom.
2. GROUP BY customer\_id, customer\_name, order\_date panna daily uniqueness maintain pannrom.
3. COUNT(\*) AS late\_days panna romba late order days count pannrom.
4. WHERE late\_days = 10 vechutu 10 continuous late night order pannra customers identify pannrom.

### Eg:

* + Aravind 10 days continuous late night orders pannirukkanga ✅ will be shown.
  + Selvi 10 days continuous la illa ❌ will not be shown.

## 🍔 Use Case Value (Tanglish)

Swiggy and Gowtham can use this to identify **late night foodies** for exclusive night meal offers and targeted delivery promotions.

# 📦 Problem 9: Amazon – High Spend Customers

## 🧩 Question (Tanglish)

Gowtham wants to find Amazon customers (like Lakshmi, Kiran) who spent more than ₹1 Lakh in the last year.

## 🗂 Problem Statement (Tanglish)

Amazon-la last 1 year-la 1 Lakh mela spend pannina customers identify panna vendiyathu for premium customer tagging.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_orders\_high\_spend ( order\_id INT PRIMARY KEY,

customer\_id INT, customer\_name VARCHAR(50), order\_amount DECIMAL(10,2), order\_date DATE

);

INSERT INTO amazon\_orders\_high\_spend VALUES (1, 1001, 'Lakshmi', 25000.00, '2024-08-01'),

(2, 1001, 'Lakshmi', 30000.00, '2024-10-12'),

(3, 1001, 'Lakshmi', 50000.00, '2025-03-05'),

(4, 1002, 'Kiran', 20000.00, '2024-09-15');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, SUM(order\_amount) AS total\_spent FROM amazon\_orders\_high\_spend

WHERE order\_date >= DATE\_SUB(CURDATE(), INTERVAL 1 YEAR)

GROUP BY customer\_id, customer\_name HAVING total\_spent > 100000;

## 🩻 Explanation (Tanglish)

### ✅ Step-by-Step:

1. WHERE order\_date >= DATE\_SUB(CURDATE(), INTERVAL 1 YEAR) vechutu last 1 year orders filter pannrom.
2. SUM(order\_amount) panna total spent calculate pannrom.
3. HAVING total\_spent > 100000 vechutu 1 Lakh mela spend pannina customers identify pannrom.

### Eg:

* + Lakshmi ₹1.05L spend pannirukkanga ✅ will be shown.
  + Kiran ₹20K mattum spend pannirukkanga ❌ will not be shown.

## 📦 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **high-value customers** for premium membership, early delivery benefits, and personalized campaigns.

# 🎬 Problem 10: Netflix – Tamil Content Watchers

## 🧩 Question (Tanglish)

Gowtham wants to find Netflix users (like Harish, Kavya) who watched only Tamil content for 6 months continuously.

## 🗂 Problem Statement (Tanglish)

Netflix-la yaar 6 months continuous-ah Tamil content mattum watch pannirukkangalo, andha users identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE netflix\_tamil\_watch ( watch\_id INT PRIMARY KEY, user\_id INT,

user\_name VARCHAR(50), content\_language VARCHAR(20), watch\_date DATE

);

INSERT INTO netflix\_tamil\_watch VALUES (1, 1101, 'Harish', 'Tamil', '2025-01-10'),

(2, 1101, 'Harish', 'Tamil', '2025-02-12'),

(3, 1101, 'Harish', 'Tamil', '2025-03-05'),

(4, 1101, 'Harish', 'Tamil', '2025-04-15'),

(5, 1101, 'Harish', 'Tamil', '2025-05-20'),

(6, 1101, 'Harish', 'Tamil', '2025-06-02'),

(7, 1102, 'Kavya', 'English', '2025-03-12');

## 💡 Solution (MySQL)

WITH monthly\_tamil AS (

SELECT user\_id, user\_name, DATE\_FORMAT(watch\_date, '%Y-%m') AS watch\_month, COUNT(\*) AS tamil\_count

FROM netflix\_tamil\_watch

WHERE content\_language = 'Tamil'

GROUP BY user\_id, user\_name, watch\_month

),

month\_counts AS (

SELECT user\_id, user\_name, COUNT(DISTINCT watch\_month) AS months\_watched FROM monthly\_tamil

GROUP BY user\_id, user\_name

)

SELECT user\_id, user\_name FROM month\_counts

WHERE months\_watched = 6;

## 🩻 Explanation (Tanglish)

### ✅ Step-by-Step:

1. WHERE content\_language = 'Tamil' vechutu Tamil content mattum filter pannrom.
2. DATE\_FORMAT(watch\_date, '%Y-%m') vechutu month-wise track pannrom.
3. COUNT(DISTINCT watch\_month) panna how many months Tamil content watch pannangannu count pannrom.
4. WHERE months\_watched = 6 vechutu 6 months continuous watch pannina users identify pannrom.

### Eg:

* + Harish 6 months Tamil content watch pannirukkanga ✅ will be shown.
  + Kavya English content watch pannirukkanga ❌ will not be shown.

## 🎬 Use Case Value (Tanglish)

Netflix and Gowtham can use this to identify **Tamil loyal viewers** for regional content campaigns and personalized recommendations.

# 🍔 Problem 11: Zomato – Festival Orders Tracker

## 🧩 Question (Tanglish)

Gowtham wants to find Zomato customers (like Nisha, Rajesh) who ordered sweets during all major festivals in the last year.

## 🗂 Problem Statement (Tanglish)

Zomato-la last 1 year festivals-la sweets order pannina consistent customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE zomato\_festival\_orders ( order\_id INT PRIMARY KEY, customer\_id INT,

customer\_name VARCHAR(50), item\_category VARCHAR(20), order\_date DATE

);

INSERT INTO zomato\_festival\_orders VALUES (1, 1201, 'Nisha', 'Sweet', '2024-10-24'),

(2, 1201, 'Nisha', 'Sweet', '2025-01-14'),

(3, 1201, 'Nisha', 'Sweet', '2025-03-29'),

(4, 1202, 'Rajesh', 'Sweet', '2024-10-24'),

(5, 1202, 'Rajesh', 'Biryani', '2025-01-14');

## 💡 Solution (MySQL)

WITH festival\_orders AS (

SELECT customer\_id, customer\_name, COUNT(DISTINCT order\_date) AS festival\_days FROM zomato\_festival\_orders

WHERE item\_category = 'Sweet'

GROUP BY customer\_id, customer\_name

)

SELECT customer\_id, customer\_name FROM festival\_orders

WHERE festival\_days >= 3;

## 🩻 Explanation (Tanglish)

✅ Sweet orders mattum filter panni, festival dates la count panni, 3 festivals la sweets order pannina customers identify pannrom.

**Eg:** Nisha 3 festivals la order pannirukkanga ✅, Rajesh only 1 festival mattum pannirukkanga

❌.

## 🍔 Use Case Value (Tanglish)

Zomato and Gowtham can use this to identify festival-order customers for special campaign offers.

# 🚖 Problem 12: Uber – Multi-City Riders

## 🧩 Question (Tanglish)

Gowtham wants to find Uber customers (like Vignesh, Sneha) who took rides in more than 3 different cities in the last 6 months.

## 🗂 Problem Statement (Tanglish)

Uber-la last 6 months la 3 city mela rides edukka consistent customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE uber\_multi\_city ( ride\_id INT PRIMARY KEY, customer\_id INT, customer\_name VARCHAR(50), city VARCHAR(50),

ride\_date DATE

);

INSERT INTO uber\_multi\_city VALUES

(1, 1301, 'Vignesh', 'Chennai', '2025-02-10'),

(2, 1301, 'Vignesh', 'Bangalore', '2025-03-15'),

(3, 1301, 'Vignesh', 'Coimbatore', '2025-04-20'),

(4, 1301, 'Vignesh', 'Hyderabad', '2025-05-25'),

(5, 1302, 'Sneha', 'Chennai', '2025-03-10');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(DISTINCT city) AS cities\_visited FROM uber\_multi\_city

WHERE ride\_date >= DATE\_SUB(CURDATE(), INTERVAL 6 MONTH)

GROUP BY customer\_id, customer\_name HAVING cities\_visited > 3;

## 🩻 Explanation (Tanglish)

✅ Last 6 months rides filter panni, different cities count panni, 3 mela city la ride edukka customers identify pannrom.

**Eg:** Vignesh 4 city la ride pannirukkanga ✅, Sneha 1 city mattum pannirukkanga ❌.

## 🚖 Use Case Value (Tanglish)

Uber and Gowtham can use this to identify **multi-city frequent travellers** for premium offers.

# 📦 Problem 13: Amazon – Coupon Usage Tracker

## 🧩 Question (Tanglish)

Gowtham wants to find Amazon customers (like Anitha, Manoj) who used discount coupons in all their purchases in the last 3 months.

## 🗂 Problem Statement (Tanglish)

Amazon-la last 3 months la every order-ku coupon use pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_coupon\_usage ( order\_id INT PRIMARY KEY, customer\_id INT,

customer\_name VARCHAR(50), coupon\_used BOOLEAN, order\_date DATE

);

INSERT INTO amazon\_coupon\_usage VALUES (1, 1401, 'Anitha', TRUE, '2025-04-02'),

(2, 1401, 'Anitha', TRUE, '2025-05-10'),

(3, 1401, 'Anitha', TRUE, '2025-06-12'),

(4, 1402, 'Manoj', TRUE, '2025-04-15'),

(5, 1402, 'Manoj', FALSE, '2025-05-20');

## 💡 Solution (MySQL)

WITH total\_orders AS (

SELECT customer\_id, customer\_name, COUNT(\*) AS total\_orders FROM amazon\_coupon\_usage

WHERE order\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

GROUP BY customer\_id, customer\_name

),

coupon\_orders AS (

SELECT customer\_id, customer\_name, COUNT(\*) AS coupon\_orders FROM amazon\_coupon\_usage

WHERE coupon\_used = TRUE AND order\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

GROUP BY customer\_id, customer\_name

)

SELECT t.customer\_id, t.customer\_name FROM total\_orders t

JOIN coupon\_orders c ON t.customer\_id = c.customer\_id WHERE t.total\_orders = c.coupon\_orders;

## 🩻 Explanation (Tanglish)

✅ Last 3 months la orders filter panni, total orders count pannrom, coupon used orders count pannrom, both equal na customers identify pannrom.

**Eg:** Anitha all orders ku coupon use pannirukkanga ✅, Manoj oru order la coupon use pannala

❌.

## 📦 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **coupon dependent customers** for exclusive discount campaigns and retention offers.

# 🍔 Problem 14: Zomato – Weekly Consistent Order Customers

## 🧩 Question (Tanglish)

Gowtham wants to find Zomato customers (like Santhosh, Keerthi) who ordered at least once every week for the last 8 weeks.

## 🗂 Problem Statement (Tanglish)

Zomato-la last 8 weeks la weekly atleast 1 order pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE zomato\_weekly\_orders ( order\_id INT PRIMARY KEY, customer\_id INT,

customer\_name VARCHAR(50), order\_date DATE

);

INSERT INTO zomato\_weekly\_orders VALUES (1, 1501, 'Santhosh', '2025-05-01'),

(2, 1501, 'Santhosh', '2025-05-08'),

(3, 1501, 'Santhosh', '2025-05-15'),

(4, 1501, 'Santhosh', '2025-05-22'),

(5, 1501, 'Santhosh', '2025-05-29'),

(6, 1501, 'Santhosh', '2025-06-05'),

(7, 1501, 'Santhosh', '2025-06-12'),

(8, 1501, 'Santhosh', '2025-06-19'),

(9, 1502, 'Keerthi', '2025-05-01');

## 💡 Solution (MySQL)

WITH weekly\_orders AS (

SELECT customer\_id, customer\_name, YEARWEEK(order\_date) AS year\_week FROM zomato\_weekly\_orders

WHERE order\_date >= DATE\_SUB(CURDATE(), INTERVAL 8 WEEK)

GROUP BY customer\_id, customer\_name, year\_week

)

SELECT customer\_id, customer\_name FROM weekly\_orders

GROUP BY customer\_id, customer\_name HAVING COUNT(DISTINCT year\_week) = 8;

## 🩻 Explanation (Tanglish)

✅ Last 8 weeks data filter panni, weekly atleast 1 order panna customers identify panna

YEARWEEK use pannrom. 8 unique weeks irundhaa show pannrom.

**Eg:** Santhosh 8 weeks la order pannirukkanga ✅, Keerthi 1 week mattum pannirukkanga ❌.

## 🍔 Use Case Value (Tanglish)

Zomato and Gowtham can use this to identify **weekly loyal customers** for exclusive delivery benefits and rewards.

# 🚖 Problem 15: Uber – Cash Payment Consistency

## 🧩 Question (Tanglish)

Gowtham wants to find Uber customers (like Muthu, Priya) who paid cash for all rides in the last 3 months.

## 🗂 Problem Statement (Tanglish)

Uber-la last 3 months la ella rides-um cash la pay pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE uber\_cash\_payments ( ride\_id INT PRIMARY KEY, customer\_id INT,

customer\_name VARCHAR(50), payment\_mode VARCHAR(20), ride\_date DATE

);

INSERT INTO uber\_cash\_payments VALUES (1, 1601, 'Muthu', 'Cash', '2025-04-05'),

(2, 1601, 'Muthu', 'Cash', '2025-05-10'),

(3, 1601, 'Muthu', 'Cash', '2025-06-15'),

(4, 1602, 'Priya', 'Card', '2025-04-08');

## 💡 Solution (MySQL)

WITH total\_rides AS (

SELECT customer\_id, customer\_name, COUNT(\*) AS total\_rides FROM uber\_cash\_payments

WHERE ride\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

GROUP BY customer\_id, customer\_name

),

cash\_rides AS (

SELECT customer\_id, customer\_name, COUNT(\*) AS cash\_rides FROM uber\_cash\_payments

WHERE payment\_mode = 'Cash' AND ride\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

GROUP BY customer\_id, customer\_name

)

SELECT t.customer\_id, t.customer\_name FROM total\_rides t

JOIN cash\_rides c ON t.customer\_id = c.customer\_id WHERE t.total\_rides = c.cash\_rides;

## 🩻 Explanation (Tanglish)

✅ Last 3 months rides filter panni, total rides count pannrom, cash rides count pannrom, rendu equal aa irundhaa show pannrom.

**Eg:** Muthu all rides cash la pay pannirukkanga ✅, Priya card use pannirukkanga ❌.

## 🚖 Use Case Value (Tanglish)

Uber and Gowtham can use this to identify **cash-only users** for cash payment offers and risk evaluation.

# 🎬 Problem 16: Netflix – Genre Diversity Watchers

## 🧩 Question (Tanglish)

Gowtham wants to find Netflix users (like Harini, Ajay) who watched content from at least 5 different genres in the last 2 months.

## 🗂 Problem Statement (Tanglish)

Netflix-la last 2 months la 5 different genres watch pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE netflix\_genre\_watch ( watch\_id INT PRIMARY KEY, user\_id INT,

user\_name VARCHAR(50), genre VARCHAR(30),

watch\_date DATE

);

INSERT INTO netflix\_genre\_watch VALUES (1, 1701, 'Harini', 'Action', '2025-05-02'),

(2, 1701, 'Harini', 'Drama', '2025-05-10'),

(3, 1701, 'Harini', 'Comedy', '2025-05-15'),

(4, 1701, 'Harini', 'Thriller', '2025-06-05'),

(5, 1701, 'Harini', 'Romance', '2025-06-10'),

(6, 1702, 'Ajay', 'Action', '2025-06-12');

## 💡 Solution (MySQL)

SELECT user\_id, user\_name, COUNT(DISTINCT genre) AS genre\_count FROM netflix\_genre\_watch

WHERE watch\_date >= DATE\_SUB(CURDATE(), INTERVAL 2 MONTH)

GROUP BY user\_id, user\_name HAVING genre\_count >= 5;

## 🩻 Explanation (Tanglish)

✅ Last 2 months la data filter pannrom, genre wise count pannrom, 5 genres cross pannina customers identify pannrom.

**Eg:** Harini 5 genres watch pannirukkanga ✅, Ajay only 1 genre pannirukkanga ❌.

## 🎬 Use Case Value (Tanglish)

Netflix and Gowtham can use this to identify **diverse genre viewers** for personalized recommendations.

# 📱 Problem 17: Meta – Daily Story Uploaders

## 🧩 Question (Tanglish)

Gowtham wants to find Meta users (like Arun, Divya) who uploaded stories daily for 30 consecutive days.

## 🗂 Problem Statement (Tanglish)

Meta-la 30 days continuous-a daily story upload pannina users identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE meta\_story\_uploads ( story\_id INT PRIMARY KEY, user\_id INT,

user\_name VARCHAR(50), upload\_date DATE

);

INSERT INTO meta\_story\_uploads VALUES (1, 1801, 'Arun', '2025-05-01'),

(2, 1801, 'Arun', '2025-05-02'),

-- (add 28 more for Arun)

(31, 1802, 'Divya', '2025-05-01');

## 💡 Solution (MySQL)

SELECT user\_id, user\_name, COUNT(DISTINCT upload\_date) AS upload\_days FROM meta\_story\_uploads

WHERE upload\_date >= DATE\_SUB(CURDATE(), INTERVAL 30 DAY)

GROUP BY user\_id, user\_name HAVING upload\_days = 30;

## 🩻 Explanation (Tanglish)

✅ Last 30 days la daily story upload pannina users identify panna COUNT DISTINCT dates use pannrom. 30 days match aagum pothu show pannrom.

**Eg:** Arun 30 days continuous story upload pannirukkanga ✅, Divya only 1 day pannirukkanga

❌.

## 📱 Use Case Value (Tanglish)

Meta and Gowtham can use this to identify **highly active daily story users** for exclusive features and engagement rewards.

# 🚖 Problem 18: Uber – Late Night Frequent Riders

## 🧩 Question (Tanglish)

Gowtham wants to find Uber customers (like Saravanan, Meera) who took rides after 10 PM at least 20 times in the last 2 months.

## 🗂 Problem Statement (Tanglish)

Uber-la last 2 months la 10 PM ku appuram atleast 20 rides edutha customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE uber\_late\_rides ( ride\_id INT PRIMARY KEY, customer\_id INT, customer\_name VARCHAR(50), ride\_time TIME,

ride\_date DATE

);

INSERT INTO uber\_late\_rides VALUES

(1, 1901, 'Saravanan', '22:30:00', '2025-05-05'),

(2, 1901, 'Saravanan', '23:00:00', '2025-05-06'),

-- (add 18 more for Saravanan)

(21, 1902, 'Meera', '21:00:00', '2025-05-05');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS late\_rides FROM uber\_late\_rides

WHERE ride\_time >= '22:00:00' AND ride\_date >= DATE\_SUB(CURDATE(), INTERVAL 2 MONTH)

GROUP BY customer\_id, customer\_name HAVING late\_rides >= 20;

## 🩻 Explanation (Tanglish)

✅ Last 2 months la 10 PM mela rides filter pannrom, count pannitu 20 rides cross pannina customers identify pannrom.

**Eg:** Saravanan 20 rides pannirukkanga ✅, Meera 10 PM ku mela illa ❌.

## 🚖 Use Case Value (Tanglish)

Uber and Gowtham can use this to identify **late night frequent riders** for targeted offers.

# 🍔 Problem 19: Swiggy – Bulk Order Customers

## 🧩 Question (Tanglish)

Gowtham wants to find Swiggy customers (like Kumar, Anjali) who placed orders above ₹1000 at least 5 times in the last 3 months.

## 🗂 Problem Statement (Tanglish)

Swiggy-la last 3 months la ₹1000 mela order 5 times panna customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE swiggy\_bulk\_orders ( order\_id INT PRIMARY KEY, customer\_id INT,

customer\_name VARCHAR(50), amount DECIMAL(10,2), order\_date DATE

);

INSERT INTO swiggy\_bulk\_orders VALUES (1, 2001, 'Kumar', 1200.00, '2025-04-01'),

(2, 2001, 'Kumar', 1500.00, '2025-04-15'),

-- (add 3 more for Kumar)

(6, 2002, 'Anjali', 800.00, '2025-04-10');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS big\_orders

FROM swiggy\_bulk\_orders

WHERE amount >= 1000 AND order\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

GROUP BY customer\_id, customer\_name HAVING big\_orders >= 5;

## 🩻 Explanation (Tanglish)

✅ Last 3 months data filter panni, ₹1000 mela order count pannitu, 5 times cross pannina customers identify pannrom.

**Eg:** Kumar 5 big orders pannirukkanga ✅, Anjali ₹1000 mela illa ❌.

## 🍔 Use Case Value (Tanglish)

Swiggy and Gowtham can use this to identify **bulk order customers** for catering promotions.

# 🎬 Problem 20: Netflix – Binge Watch Series Finishers

## 🧩 Question (Tanglish)

Gowtham wants to find Netflix users (like Rahul, Sneha) who completed an entire series within 2 days.

## 🗂 Problem Statement (Tanglish)

Netflix-la oru series 2 days kuLLa complete pannina users identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE netflix\_series\_watch ( watch\_id INT PRIMARY KEY, user\_id INT,

user\_name VARCHAR(50), series\_id INT,

episode\_id INT, watch\_date DATE

);

INSERT INTO netflix\_series\_watch VALUES (1, 2101, 'Rahul', 301, 1, '2025-06-01'),

(2, 2101, 'Rahul', 301, 2, '2025-06-01'),

(3, 2101, 'Rahul', 301, 3, '2025-06-02'),

(4, 2102, 'Sneha', 302, 1, '2025-06-01');

## 💡 Solution (MySQL)

WITH series\_watch\_days AS (

SELECT user\_id, user\_name, series\_id, DATEDIFF(MAX(watch\_date), MIN(watch\_date)) AS days\_taken, COUNT(DISTINCT episode\_id) AS episodes\_watched

FROM netflix\_series\_watch

GROUP BY user\_id, user\_name, series\_id

)

SELECT user\_id, user\_name, series\_id FROM series\_watch\_days

WHERE days\_taken <= 1 AND episodes\_watched >= 3;

## 🩻 Explanation (Tanglish)

✅ Series wise min-max date calculate panni, difference 1 day illa adhu kuLLa irundhaa check pannrom, minimum 3 episodes irundhaa confirm pannrom.

**Eg:** Rahul 3 episodes 2 days kuLLa finish pannirukkanga ✅, Sneha only 1 episode mattum watch pannirukkanga ❌.

## 🎬 Use Case Value (Tanglish)

Netflix and Gowtham can use this to identify **binge watchers** for early content recommendations.

# 📦 Problem 21: Amazon – Return Request Trackers

## 🧩 Question (Tanglish)

Gowtham wants to find Amazon customers (like Karthik, Deepa) who raised return requests for more than 5 orders in the last 6 months.

## 🗂 Problem Statement (Tanglish)

Amazon-la last 6 months la 5 mela orders return request panna customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_returns ( return\_id INT PRIMARY KEY, customer\_id INT, customer\_name VARCHAR(50), order\_id INT,

return\_date DATE

);

INSERT INTO amazon\_returns VALUES (1, 2201, 'Karthik', 401, '2025-01-15'),

(2, 2201, 'Karthik', 402, '2025-02-20'),

-- (add 4 more for Karthik)

(7, 2202, 'Deepa', 403, '2025-03-05');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS return\_count FROM amazon\_returns

WHERE return\_date >= DATE\_SUB(CURDATE(), INTERVAL 6 MONTH)

GROUP BY customer\_id, customer\_name HAVING return\_count > 5;

## 🩻 Explanation (Tanglish)

✅ Last 6 months data filter panni, return requests count pannitu 5 cross pannina customers identify pannrom.

**Eg:** Karthik 6 returns pannirukkanga ✅, Deepa 1 return mattum pannirukkanga ❌.

## 📦 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **frequent return request customers** for policy review or proactive support.

# 📦 Problem 22: Amazon – Mobile and Accessories Together Purchasers

## 🧩 Question (Tanglish)

Gowtham wants to find Amazon customers (like Bharath, Rekha) who purchased a mobile and accessories in the same order.

## 🗂 Problem Statement (Tanglish)

Amazon-la mobile and accessories same order la vaangina customers identify panna vendiyathu for bundle offer targeting.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_orders\_bundle ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), product\_category VARCHAR(50), order\_date DATE

);

INSERT INTO amazon\_orders\_bundle VALUES (501, 2301, 'Bharath', 'Mobile', '2025-06-01'),

(501, 2301, 'Bharath', 'Accessories', '2025-06-01'),

(502, 2302, 'Rekha', 'Mobile', '2025-06-02'),

(503, 2302, 'Rekha', 'Accessories', '2025-06-03');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, order\_id FROM amazon\_orders\_bundle

WHERE product\_category IN ('Mobile', 'Accessories') GROUP BY customer\_id, customer\_name, order\_id HAVING COUNT(DISTINCT product\_category) = 2;

## 🩻 Explanation (Tanglish)

* SELECT customer\_id, customer\_name, order\_id → Customer, name, and order ID select pannrom.
* FROM amazon\_orders\_bundle → Table select pannrom.
* WHERE product\_category IN ('Mobile', 'Accessories') → Mobile and accessories rows maathiri filter pannrom.
* GROUP BY customer\_id, customer\_name, order\_id → Customer and order wise group pannrom.
* HAVING COUNT(DISTINCT product\_category) = 2 → Andha order la rendu category irukkaa nu check pannrom (mobile and accessories iruntha show aagum).

**Eg:** Bharath same order la mobile and accessories vaangirukkanga ✅, Rekha separate orders la vaangirukkanga ❌.

## 📦 Use Case Value (Tanglish)

Amazon and Gowtham can use this to target **mobile buyers with accessory bundle offers.**

# 🎬 Problem 23: Netflix – Single Day Multiple Device Watchers

## 🧩 Question (Tanglish)

Gowtham wants to find Netflix users (like Suresh, Latha) who watched content on more than 2 devices on the same day.

## 🗂 Problem Statement (Tanglish)

Netflix-la oru naal la 2 devices mela content watch pannina users identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE netflix\_device\_watch ( watch\_id INT,

user\_id INT,

user\_name VARCHAR(50), device\_name VARCHAR(50), watch\_date DATE

);

INSERT INTO netflix\_device\_watch VALUES (601, 2401, 'Suresh', 'Mobile', '2025-06-05'),

(602, 2401, 'Suresh', 'Laptop', '2025-06-05'),

(603, 2401, 'Suresh', 'TV', '2025-06-05'),

(604, 2402, 'Latha', 'Mobile', '2025-06-05');

## 💡 Solution (MySQL)

SELECT user\_id, user\_name, watch\_date FROM netflix\_device\_watch

GROUP BY user\_id, user\_name, watch\_date HAVING COUNT(DISTINCT device\_name) > 2;

## 🩻 Explanation (Tanglish)

* SELECT user\_id, user\_name, watch\_date → User, name, date select pannrom.
* FROM netflix\_device\_watch → Table select pannrom.
* GROUP BY user\_id, user\_name, watch\_date → User and date wise group pannrom.
* HAVING COUNT(DISTINCT device\_name) > 2 → Same day la 2 device mela irundhaa show pannrom.

**Eg:** Suresh 3 devices same day la watch pannirukkanga ✅, Latha 1 device mattum use pannirukkanga ❌.

## 🎬 Use Case Value (Tanglish)

Netflix and Gowtham can use this to identify **heavy multi-device users for family plan promotions.**

# 🍔 Problem 24: Swiggy – Night Owl Customers

## 🧩 Question (Tanglish)

Gowtham wants to find Swiggy customers (like Vasu, Priyanka) who ordered food after midnight at least 10 times in the last month.

## 🗂 Problem Statement (Tanglish)

Swiggy-la last month la 12 AM ku appuram 10 orders pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE swiggy\_night\_orders ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), order\_time TIME,

order\_date DATE

);

INSERT INTO swiggy\_night\_orders VALUES (701, 2501, 'Vasu', '00:30:00', '2025-06-01'),

(702, 2501, 'Vasu', '01:00:00', '2025-06-02'),

-- (add 8 more for Vasu)

(711, 2502, 'Priyanka', '23:50:00', '2025-06-01');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS night\_orders FROM swiggy\_night\_orders

WHERE order\_time >= '00:00:00' AND order\_time < '06:00:00' AND order\_date >= DATE\_SUB(CURDATE(), INTERVAL 1 MONTH)

GROUP BY customer\_id, customer\_name

HAVING night\_orders >= 10;

## 🩻 Explanation (Tanglish)

* SELECT customer\_id, customer\_name, COUNT(\*) AS night\_orders → Customer, name, and night order count select pannrom.
* FROM swiggy\_night\_orders → Table select pannrom.
* WHERE order\_time >= '00:00:00' AND order\_time < '06:00:00' → Midnight to 6 AM orders filter pannrom.
* AND order\_date >= DATE\_SUB(CURDATE(), INTERVAL 1 MONTH) → Last 1

month data filter pannrom.

* GROUP BY customer\_id, customer\_name → Customer wise group pannrom.
* HAVING night\_orders >= 10 → 10 orders cross pannina customers show pannrom.

**Eg:** Vasu 10 night orders pannirukkanga ✅, Priyanka 12 AM ku appuram illa ❌.

## 🍔 Use Case Value (Tanglish)

Swiggy and Gowtham can use this to identify **night owl customers for special midnight deals.**

# 📦 Problem 25: Amazon – Customers Buying During Flash Sales

## 🧩 Question (Tanglish)

Gowtham wants to find Amazon customers (like Arjun, Divya) who purchased products during flash sales in the last 3 months.

## 🗂 Problem Statement (Tanglish)

Amazon-la last 3 months flash sale time la purchase pannina customers identify panna vendiyathu for special loyalty tagging.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_flash\_sales ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), order\_datetime DATETIME

);

INSERT INTO amazon\_flash\_sales VALUES (801, 2601, 'Arjun', '2025-06-01 12:05:00'),

(802, 2601, 'Arjun', '2025-06-15 12:10:00'),

(803, 2602, 'Divya', '2025-06-01 14:00:00');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS flash\_orders FROM amazon\_flash\_sales

WHERE TIME(order\_datetime) BETWEEN '12:00:00' AND '12:30:00' AND order\_datetime >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

GROUP BY customer\_id, customer\_name;

## 🩻 Explanation (Tanglish)

* SELECT customer\_id, customer\_name, COUNT(\*) AS flash\_orders → Customer, name, flash sale orders count select pannrom.
* FROM amazon\_flash\_sales → Table select pannrom.
* WHERE TIME(order\_datetime) BETWEEN '12:00:00' AND '12:30:00' →

Flash sale time window la irukka filter pannrom.

* AND order\_datetime >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH) → Last 3 months data filter pannrom.
* GROUP BY customer\_id, customer\_name → Customer wise group pannrom for flash sale tracking.

**Eg:** Arjun 2 flash sale orders pannirukkanga ✅, Divya flash window time la illa ❌.

## 📦 Use Case Value (Tanglish)

Amazon and Gowtham can use this for **flash sale loyal customer targeting and reward programs.**

# 🎬 Problem 26: Netflix – Users Watching During Weekend Nights

## 🧩 Question (Tanglish)

Gowtham wants to find Netflix users (like Mani, Sneha) who watched movies during weekend nights (Saturday, Sunday after 9 PM) at least 8 times in the last 2 months.

## 🗂 Problem Statement (Tanglish)

Netflix-la last 2 months la Saturday, Sunday 9 PM ku mela 8 times movies watch pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE netflix\_weekend\_watch ( watch\_id INT,

user\_id INT,

user\_name VARCHAR(50), watch\_datetime DATETIME

);

INSERT INTO netflix\_weekend\_watch VALUES (901, 2701, 'Mani', '2025-06-07 21:30:00'),

(902, 2701, 'Mani', '2025-06-08 22:00:00'),

-- (add 6 more for Mani)

(910, 2702, 'Sneha', '2025-06-07 20:00:00');

## 💡 Solution (MySQL)

SELECT user\_id, user\_name, COUNT(\*) AS weekend\_night\_views FROM netflix\_weekend\_watch

WHERE DAYOFWEEK(watch\_datetime) IN (1,7) AND TIME(watch\_datetime) >= '21:00:00'

AND watch\_datetime >= DATE\_SUB(CURDATE(), INTERVAL 2 MONTH)

GROUP BY user\_id, user\_name HAVING weekend\_night\_views >= 8;

## 🩻 Explanation (Tanglish)

* SELECT user\_id, user\_name, COUNT(\*) AS weekend\_night\_views → User, name, weekend night views count select pannrom.
* FROM netflix\_weekend\_watch → Table select pannrom.
* WHERE DAYOFWEEK(watch\_datetime) IN (1,7) → Saturday (7), Sunday (1) filter pannrom.
* AND TIME(watch\_datetime) >= '21:00:00' → 9 PM ku mela irukka filter pannrom.
* AND watch\_datetime >= DATE\_SUB(CURDATE(), INTERVAL 2 MONTH) → Last 2 months data filter pannrom.
* GROUP BY user\_id, user\_name → User wise group pannrom.
* HAVING weekend\_night\_views >= 8 → 8 views cross pannina users select pannrom.

**Eg:** Mani 8 weekend night views pannirukkanga ✅, Sneha time mismatch aagiduchu ❌.

## 🎬 Use Case Value (Tanglish)

Netflix and Gowtham can use this to identify **weekend binge watchers for night plan offers and recommendations.**

# 🚖 Problem 27: Uber – Long Distance Riders

## 🧩 Question (Tanglish)

Gowtham wants to find Uber customers (like Raja, Kavitha) who completed rides longer than 30 km at least 5 times in the last 3 months.

## 🗂 Problem Statement (Tanglish)

Uber-la last 3 months la 30 km mela ride 5 times pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE uber\_long\_rides ( ride\_id INT,

customer\_id INT, customer\_name VARCHAR(50), distance\_km DECIMAL(5,2), ride\_date DATE

);

INSERT INTO uber\_long\_rides VALUES (1001, 2801, 'Raja', 35.5, '2025-04-01'),

(1002, 2801, 'Raja', 40.0, '2025-05-05'),

-- (add 3 more for Raja)

(1006, 2802, 'Kavitha', 25.0, '2025-04-10');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS long\_ride\_count FROM uber\_long\_rides

WHERE distance\_km > 30 AND ride\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH) GROUP BY customer\_id, customer\_name

HAVING long\_ride\_count >= 5;

## 🩻 Explanation (Tanglish)

* SELECT customer\_id, customer\_name, COUNT(\*) AS long\_ride\_count → Customer, name, long ride count select pannrom.
* FROM uber\_long\_rides → Table select pannrom.
* WHERE distance\_km > 30 → 30 km mela irukka filter pannrom.
* AND ride\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH) → Last 3

months data filter pannrom.

* GROUP BY customer\_id, customer\_name → Customer wise group pannrom.
* HAVING long\_ride\_count >= 5 → 5 times cross pannina customers identify pannrom.

**Eg:** Raja 5 long rides pannirukkanga ✅, Kavitha distance kammiya irukku ❌.

## 🚖 Use Case Value (Tanglish)

Uber and Gowtham can use this for **long distance user rewards and personalized pricing.**

# 📦 Problem 28: Amazon – Consistent Monthly Purchasers

## 🧩 Question (Tanglish)

Gowtham wants to find Amazon customers (like Prakash, Swathi) who made at least 1 purchase every month consistently for the last 6 months.

## 🗂 Problem Statement (Tanglish)

Amazon-la last 6 months la monthly atleast 1 purchase pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_monthly\_purchases ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), order\_date DATE

);

INSERT INTO amazon\_monthly\_purchases VALUES (1101, 2901, 'Prakash', '2025-01-15'),

(1102, 2901, 'Prakash', '2025-02-10'),

(1103, 2901, 'Prakash', '2025-03-05'),

(1104, 2901, 'Prakash', '2025-04-20'),

(1105, 2901, 'Prakash', '2025-05-18'),

(1106, 2901, 'Prakash', '2025-06-12'),

(1107, 2902, 'Swathi', '2025-02-10');

## 💡 Solution (MySQL)

WITH monthwise AS (

SELECT customer\_id, customer\_name, DATE\_FORMAT(order\_date, '%Y-%m') AS order\_month

FROM amazon\_monthly\_purchases

WHERE order\_date >= DATE\_SUB(CURDATE(), INTERVAL 6 MONTH)

GROUP BY customer\_id, customer\_name, order\_month

)

SELECT customer\_id, customer\_name, COUNT(DISTINCT order\_month) AS active\_months FROM monthwise

GROUP BY customer\_id, customer\_name HAVING active\_months = 6;

## 🩻 Explanation (Tanglish)

* WITH monthwise AS (...) → Last 6 months data extract panni month-wise group panna CTE create pannrom.
* DATE\_FORMAT(order\_date, '%Y-%m') → Month format la convert pannrom.
* GROUP BY customer\_id, customer\_name, order\_month → Customer and month wise unique data eduthukkrom.
* SELECT customer\_id, customer\_name, COUNT(DISTINCT order\_month) AS active\_months → Customer wise unique months count pannrom.
* HAVING active\_months = 6 → 6 months continuous purchase pannina customers show pannrom.

**Eg:** Prakash 6 months continuous purchase pannirukkanga ✅, Swathi 1 month mattum pannirukkanga ❌.

## 📦 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **consistent monthly buyers for loyalty programs and targeted marketing.**

# 🚖 Problem 29: Uber – Frequent Cancellers

## 🧩 Question (Tanglish)

Gowtham wants to find Uber customers (like Gopi, Anitha) who cancelled more than 3 rides in any month in the last 6 months.

## 🗂 Problem Statement (Tanglish)

Uber-la last 6 months la oru month la 3 mela rides cancel pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE uber\_cancellations ( ride\_id INT,

customer\_id INT, customer\_name VARCHAR(50), cancel\_flag BOOLEAN, ride\_date DATE

);

INSERT INTO uber\_cancellations VALUES (1201, 3001, 'Gopi', TRUE, '2025-04-01'),

(1202, 3001, 'Gopi', TRUE, '2025-04-02'),

(1203, 3001, 'Gopi', TRUE, '2025-04-03'),

(1204, 3001, 'Gopi', TRUE, '2025-04-04'),

(1205, 3002, 'Anitha', TRUE, '2025-05-10');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, DATE\_FORMAT(ride\_date, '%Y-%m') AS cancel\_month, COUNT(\*) AS cancel\_count

FROM uber\_cancellations

WHERE cancel\_flag = TRUE AND ride\_date >= DATE\_SUB(CURDATE(), INTERVAL 6 MONTH)

GROUP BY customer\_id, customer\_name, cancel\_month HAVING cancel\_count > 3;

## 🩻 Explanation (Tanglish)

* SELECT customer\_id, customer\_name, DATE\_FORMAT(ride\_date,

'%Y-%m') AS cancel\_month, COUNT(\*) AS cancel\_count → Customer, name,

month, cancellation count select pannrom.

* FROM uber\_cancellations → Table select pannrom.
* WHERE cancel\_flag = TRUE AND ride\_date >= DATE\_SUB(CURDATE(), INTERVAL 6 MONTH) → Cancelled rides filter pannrom, last 6 months data maathiri filter pannrom.
* GROUP BY customer\_id, customer\_name, cancel\_month → Customer and month wise group pannrom.
* HAVING cancel\_count > 3 → Oru month la 3 mela cancel pannina customers identify pannrom.

**Eg:** Gopi April month la 4 cancellations pannirukkanga ✅, Anitha 1 mattum panna ❌.

## 🚖 Use Case Value (Tanglish)

Uber and Gowtham can use this to identify **frequent cancellers for monitoring and policy adjustments.**

# 📦 Problem 30: Amazon – High Value Single Order Trackers

## 🧩 Question (Tanglish)

Gowtham wants to find Amazon customers (like Vijay, Kavya) who made a single purchase above ₹50,000 in the last year.

## 🗂 Problem Statement (Tanglish)

Amazon-la last 1 year la oru single order ₹50,000 mela spend pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_high\_value\_orders ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), order\_amount DECIMAL(10,2), order\_date DATE

);

INSERT INTO amazon\_high\_value\_orders VALUES (1301, 3101, 'Vijay', 60000.00, '2025-03-15'),

(1302, 3102, 'Kavya', 45000.00, '2025-02-20');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, order\_id, order\_amount FROM amazon\_high\_value\_orders

WHERE order\_amount > 50000 AND order\_date >= DATE\_SUB(CURDATE(), INTERVAL 1 YEAR);

## 🩻 Explanation (Tanglish)

* SELECT customer\_id, customer\_name, order\_id, order\_amount → Customer, name, order ID, amount select pannrom.
* FROM amazon\_high\_value\_orders → Table select pannrom.
* WHERE order\_amount > 50000 AND order\_date >= DATE\_SUB(CURDATE(), INTERVAL 1 YEAR) → Last 1 year la ₹50,000 mela irukkara orders filter pannrom.

**Eg:** Vijay order 60,000 pannirukkanga ✅, Kavya 45,000 pannirukkanga ❌.

## 📦 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **high value customers for premium benefits and personalized offers.**

# 🎬 Problem 31: Netflix – Consecutive Daily Watchers

## 🧩 Question (Tanglish)

Gowtham wants to find Netflix users (like Siva, Meena) who watched at least 1 movie daily for 15 consecutive days.

## 🗂 Problem Statement (Tanglish)

Netflix-la 15 days continuous-a daily atleast 1 movie watch pannina users identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE netflix\_daily\_watch ( watch\_id INT,

user\_id INT,

user\_name VARCHAR(50), watch\_date DATE

);

INSERT INTO netflix\_daily\_watch VALUES (1401, 3201, 'Siva', '2025-05-01'),

(1402, 3201, 'Siva', '2025-05-02'),

-- (add 13 more for Siva)

(1416, 3202, 'Meena', '2025-05-01');

💡 **Solution (MySQL)** SELECT user\_id, user\_name FROM (

SELECT user\_id, user\_name, COUNT(\*) AS consecutive\_days FROM (

SELECT user\_id, user\_name, watch\_date,

DATEDIFF(watch\_date, ROW\_NUMBER() OVER (PARTITION BY user\_id ORDER BY

watch\_date)) AS grp

FROM netflix\_daily\_watch

) x

GROUP BY user\_id, user\_name, grp HAVING consecutive\_days >= 15

) y;

## 🩻 Explanation (Tanglish)

* **Inner SELECT:** DATEDIFF(watch\_date, ROW\_NUMBER() OVER(...)) use pannitu consecutive days group identify pannrom.
* **Middle GROUP BY:** user, name, grp wise group pannitu consecutive count eduthukkrom.
* **HAVING consecutive\_days >= 15:** 15 days continuous watch pannina filter pannrom.
* **Outer SELECT:** user and name select pannrom for qualified users.

**Eg:** Siva 15 days continuous watch pannirukkanga ✅, Meena 1 day mattum pannirukkanga ❌.

## 🎬 Use Case Value (Tanglish)

Netflix and Gowtham can use this to identify **daily active binge watchers for retention strategies and early access promotions.**

# 📦 Problem 32: Amazon – Coupon Dependent Customers

## 🧩 Question (Tanglish)

Gowtham wants to find Amazon customers (like Lokesh, Sandhya) who used coupons in all their orders in the last 4 months.

## 🗂 Problem Statement (Tanglish)

Amazon-la last 4 months la eduththa ella orders-kum coupon use pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_coupon\_orders ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), coupon\_used BOOLEAN, order\_date DATE

);

INSERT INTO amazon\_coupon\_orders VALUES (1501, 3301, 'Lokesh', TRUE, '2025-03-05'),

(1502, 3301, 'Lokesh', TRUE, '2025-04-10'),

(1503, 3301, 'Lokesh', TRUE, '2025-05-15'),

(1504, 3302, 'Sandhya', TRUE, '2025-04-12'),

(1505, 3302, 'Sandhya', FALSE, '2025-05-18');

## 💡 Solution (MySQL)

WITH total\_orders AS (

SELECT customer\_id, customer\_name, COUNT(\*) AS total\_orders FROM amazon\_coupon\_orders

WHERE order\_date >= DATE\_SUB(CURDATE(), INTERVAL 4 MONTH)

GROUP BY customer\_id, customer\_name

),

coupon\_orders AS (

SELECT customer\_id, customer\_name, COUNT(\*) AS coupon\_orders FROM amazon\_coupon\_orders

WHERE coupon\_used = TRUE AND order\_date >= DATE\_SUB(CURDATE(), INTERVAL 4 MONTH)

GROUP BY customer\_id, customer\_name

)

SELECT t.customer\_id, t.customer\_name FROM total\_orders t

JOIN coupon\_orders c ON t.customer\_id = c.customer\_id WHERE t.total\_orders = c.coupon\_orders;

## 🩻 Explanation (Tanglish)

* **WITH total\_orders AS (...):** Last 4 months la eduththa total orders count pannrom.
* **WITH coupon\_orders AS (...):** Last 4 months la coupon use pannina orders mattum count pannrom.
* **JOIN:** total\_orders and coupon\_orders customer\_id moolama join pannrom.
* **WHERE t.total\_orders = c.coupon\_orders:** Total orders count and coupon used orders count equal-a irundha customers filter pannrom.

**Eg:** Lokesh ellaa orders kum coupon use pannirukkanga ✅, Sandhya oru order-la coupon use pannala ❌.

## 📦 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **coupon dependent customers for targeted coupon campaigns.**

# 🎬 Problem 33: Netflix – Multi Genre Watchers

## 🧩 Question (Tanglish)

Gowtham wants to find Netflix users (like Hari, Ramya) who watched content from at least 4 different genres in the last 2 months.

## 🗂 Problem Statement (Tanglish)

Netflix-la last 2 months la 4 genres mela content watch pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE netflix\_genre\_watchers ( watch\_id INT,

user\_id INT,

user\_name VARCHAR(50), genre VARCHAR(30),

watch\_date DATE

);

INSERT INTO netflix\_genre\_watchers VALUES (1601, 3401, 'Hari', 'Action', '2025-05-01'),

(1602, 3401, 'Hari', 'Drama', '2025-05-10'),

(1603, 3401, 'Hari', 'Comedy', '2025-05-15'),

(1604, 3401, 'Hari', 'Thriller', '2025-06-01'),

(1605, 3402, 'Ramya', 'Action', '2025-05-05');

## 💡 Solution (MySQL)

SELECT user\_id, user\_name, COUNT(DISTINCT genre) AS genre\_count FROM netflix\_genre\_watchers

WHERE watch\_date >= DATE\_SUB(CURDATE(), INTERVAL 2 MONTH)

GROUP BY user\_id, user\_name

HAVING genre\_count >= 4;

## 🩻 Explanation (Tanglish)

* **SELECT user\_id, user\_name, COUNT(DISTINCT genre) AS genre\_count:** User-wise genre diversity count pannrom.
* **FROM netflix\_genre\_watchers:** Table select pannrom.

### WHERE watch\_date >= DATE\_SUB(CURDATE(), INTERVAL 2 MONTH): Last 2

months data filter pannrom.

* **GROUP BY user\_id, user\_name:** User-wise group pannrom.
* **HAVING genre\_count >= 4:** 4 genres watch pannina users filter pannrom.

**Eg:** Hari 4 genres watch pannirukkanga ✅, Ramya 1 genre mattum pannirukkanga ❌.

## 🎬 Use Case Value (Tanglish)

Netflix and Gowtham can use this to identify **genre diverse viewers for personalized content curation.**

# 🚖 Problem 34: Uber – Inactive Users in Last Month

## 🧩 Question (Tanglish)

Gowtham wants to find Uber customers (like Rohit, Sneha) who did not take any ride in the last 1 month.

## 🗂 Problem Statement (Tanglish)

Uber-la last 1 month la oru ride-um edukkaama inactive-a irundha customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE uber\_rides\_activity ( ride\_id INT,

customer\_id INT, customer\_name VARCHAR(50), ride\_date DATE

);

INSERT INTO uber\_rides\_activity VALUES (1701, 3501, 'Rohit', '2025-04-15'),

(1702, 3502, 'Sneha', '2025-06-01');

## 💡 Solution (MySQL)

SELECT DISTINCT customer\_id, customer\_name FROM uber\_rides\_activity

WHERE customer\_id NOT IN ( SELECT DISTINCT customer\_id FROM uber\_rides\_activity

WHERE ride\_date >= DATE\_SUB(CURDATE(), INTERVAL 1 MONTH)

);

## 🩻 Explanation (Tanglish)

* **Inner SELECT:** Last 1 month la ride edutha customers select pannrom.
* **Outer SELECT:** andha customers illadha customer\_id matta select pannrom.
* **DISTINCT:** duplicate records varakoodaadhu nu prevent pannrom.

**Eg:** Rohit last 1 month la ride edukkaala ✅, Sneha ride eduthirukkanga ❌.

## 🚖 Use Case Value (Tanglish)

Uber and Gowtham can use this to identify **inactive users for re-engagement campaigns.**

# 📦 Problem 35: Amazon – Electronics & Fashion Combo Buyers

## 🧩 Question (Tanglish)

Gowtham wants to find Amazon customers (like Ramesh, Priya) who purchased both electronics and fashion products in the same month.

## 🗂 Problem Statement (Tanglish)

Amazon-la orae month la electronics-um fashion-um vaangina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_category\_purchases ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), category VARCHAR(30), order\_date DATE

);

INSERT INTO amazon\_category\_purchases VALUES (1801, 3601, 'Ramesh', 'Electronics', '2025-05-05'),

(1802, 3601, 'Ramesh', 'Fashion', '2025-05-15'),

(1803, 3602, 'Priya', 'Electronics', '2025-05-10');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, DATE\_FORMAT(order\_date, '%Y-%m') AS order\_month

FROM amazon\_category\_purchases WHERE category IN ('Electronics', 'Fashion')

GROUP BY customer\_id, customer\_name, order\_month HAVING COUNT(DISTINCT category) = 2;

## 🩻 Explanation (Tanglish)

* **WHERE category IN ('Electronics', 'Fashion')** → Iru category la irukka filter pannrom.
* **DATE\_FORMAT(order\_date, '%Y-%m')** → Month wise extract pannrom.
* **GROUP BY customer\_id, customer\_name, order\_month** → Customer and month wise group pannrom.
* **HAVING COUNT(DISTINCT category) = 2** → Andha month la rendu category irundha users select pannrom.

**Eg:** Ramesh same month la Electronics & Fashion vaangirukkanga ✅, Priya fashion vaangala

❌.

## 📦 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **combo buyers for cross-sell and upsell offers.**

# 🎬 Problem 36: Netflix – Heavy Reviewers

## 🧩 Question (Tanglish)

Gowtham wants to find Netflix users (like Aravind, Divya) who reviewed at least 20 movies in the last 3 months.

## 🗂 Problem Statement (Tanglish)

Netflix-la last 3 months la 20 movies ku mela review kudutha users identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE netflix\_reviews ( review\_id INT,

user\_id INT,

user\_name VARCHAR(50), movie\_id INT,

review\_date DATE

);

INSERT INTO netflix\_reviews VALUES (1901, 3701, 'Aravind', 501, '2025-04-01'),

(1902, 3701, 'Aravind', 502, '2025-04-02'),

-- (add 18 more for Aravind)

(1921, 3702, 'Divya', 503, '2025-05-10');

## 💡 Solution (MySQL)

SELECT user\_id, user\_name, COUNT(\*) AS review\_count FROM netflix\_reviews

WHERE review\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

GROUP BY user\_id, user\_name HAVING review\_count >= 20;

## 🩻 Explanation (Tanglish)

* **SELECT user\_id, user\_name, COUNT(\*) AS review\_count:** User and review count select pannrom.
* **FROM netflix\_reviews:** Table select pannrom.

### WHERE review\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH): Last 3

months data filter pannrom.

* **GROUP BY user\_id, user\_name:** User wise group pannrom.
* **HAVING review\_count >= 20:** 20 reviews cross pannina users filter pannrom.

**Eg:** Aravind 20 reviews pannirukkanga ✅, Divya 1 review mattum pannirukkanga ❌.

## 🎬 Use Case Value (Tanglish)

Netflix and Gowtham can use this to identify **heavy reviewers for advanced recommendations and beta features.**

# 🚖 Problem 37: Uber – Same Day Multi-City Travellers

## 🧩 Question (Tanglish)

Gowtham wants to find Uber customers (like Sanjay, Keerthi) who took rides in more than 1 city on the same day.

## 🗂 Problem Statement (Tanglish)

Uber-la orae naal la 2 city la ride edutha customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE uber\_multi\_city\_rides ( ride\_id INT,

customer\_id INT, customer\_name VARCHAR(50), city VARCHAR(50),

ride\_date DATE

);

INSERT INTO uber\_multi\_city\_rides VALUES (2001, 3801, 'Sanjay', 'Chennai', '2025-06-10'),

(2002, 3801, 'Sanjay', 'Bangalore', '2025-06-10'),

(2003, 3802, 'Keerthi', 'Chennai', '2025-06-11');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, ride\_date, COUNT(DISTINCT city) AS cities\_count FROM uber\_multi\_city\_rides

GROUP BY customer\_id, customer\_name, ride\_date HAVING cities\_count > 1;

## 🩻 Explanation (Tanglish)

* **SELECT customer\_id, customer\_name, ride\_date, COUNT(DISTINCT city) AS cities\_count:** Customer, name, date and unique city count select pannrom.
* **FROM uber\_multi\_city\_rides:** Table select pannrom.
* **GROUP BY customer\_id, customer\_name, ride\_date:** Customer and date wise group pannrom.
* **HAVING cities\_count > 1:** Oru naal la 2 city la ride edutha users filter pannrom.

**Eg:** Sanjay same day la 2 city la ride pannirukkanga ✅, Keerthi 1 city la mattum pannirukkanga

❌.

## 🚖 Use Case Value (Tanglish)

Uber and Gowtham can use this to identify **frequent multi-city travellers for premium ride plans.**

# 🎬 Problem 36: Netflix – Heavy Reviewers

## 🧩 Question (Tanglish)

Gowtham wants to find Netflix users (like Aravind, Divya) who reviewed at least 20 movies in the last 3 months.

## 🗂 Problem Statement (Tanglish)

Netflix-la last 3 months la 20 movies ku mela review kudutha users identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE netflix\_reviews ( review\_id INT,

user\_id INT,

user\_name VARCHAR(50), movie\_id INT,

review\_date DATE

);

INSERT INTO netflix\_reviews VALUES (1901, 3701, 'Aravind', 501, '2025-04-01'),

(1902, 3701, 'Aravind', 502, '2025-04-02'),

-- (add 18 more for Aravind)

(1921, 3702, 'Divya', 503, '2025-05-10');

## 💡 Solution (MySQL)

SELECT user\_id, user\_name, COUNT(\*) AS review\_count FROM netflix\_reviews

WHERE review\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

GROUP BY user\_id, user\_name HAVING review\_count >= 20;

## 🩻 Explanation (Tanglish)

* **SELECT user\_id, user\_name, COUNT(\*) AS review\_count:** User and review count select pannrom.
* **FROM netflix\_reviews:** Table select pannrom.

### WHERE review\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH): Last 3

months data filter pannrom.

* **GROUP BY user\_id, user\_name:** User wise group pannrom.
* **HAVING review\_count >= 20:** 20 reviews cross pannina users filter pannrom.

**Eg:** Aravind 20 reviews pannirukkanga ✅, Divya 1 review mattum pannirukkanga ❌.

## 🎬 Use Case Value (Tanglish)

Netflix and Gowtham can use this to identify **heavy reviewers for advanced recommendations and beta features.**

# 🚖 Problem 37: Uber – Same Day Multi-City Travellers

## 🧩 Question (Tanglish)

Gowtham wants to find Uber customers (like Sanjay, Keerthi) who took rides in more than 1 city on the same day.

## 🗂 Problem Statement (Tanglish)

Uber-la orae naal la 2 city la ride edutha customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE uber\_multi\_city\_rides ( ride\_id INT,

customer\_id INT, customer\_name VARCHAR(50), city VARCHAR(50),

ride\_date DATE

);

INSERT INTO uber\_multi\_city\_rides VALUES (2001, 3801, 'Sanjay', 'Chennai', '2025-06-10'),

(2002, 3801, 'Sanjay', 'Bangalore', '2025-06-10'),

(2003, 3802, 'Keerthi', 'Chennai', '2025-06-11');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, ride\_date, COUNT(DISTINCT city) AS cities\_count FROM uber\_multi\_city\_rides

GROUP BY customer\_id, customer\_name, ride\_date HAVING cities\_count > 1;

## 🩻 Explanation (Tanglish)

* **SELECT customer\_id, customer\_name, ride\_date, COUNT(DISTINCT city) AS cities\_count:** Customer, name, date and unique city count select pannrom.
* **FROM uber\_multi\_city\_rides:** Table select pannrom.
* **GROUP BY customer\_id, customer\_name, ride\_date:** Customer and date wise group pannrom.
* **HAVING cities\_count > 1:** Oru naal la 2 city la ride edutha users filter pannrom.

**Eg:** Sanjay same day la 2 city la ride pannirukkanga ✅, Keerthi 1 city la mattum pannirukkanga

❌.

## 🚖 Use Case Value (Tanglish)

Uber and Gowtham can use this to identify **frequent multi-city travellers for premium ride plans.**

# 🚖 Problem 41: Uber – Users Taking Rides with Same Driver Frequently

## 🧩 Question (Tanglish)

Gowtham wants to find Uber customers (like Karthik, Deepa) who took rides with the same driver at least 10 times in the last 6 months.

## 🗂 Problem Statement (Tanglish)

Uber-la last 6 months la orae driver kitta 10 rides edutha customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE uber\_driver\_rides ( ride\_id INT,

customer\_id INT, customer\_name VARCHAR(50), driver\_id INT,

driver\_name VARCHAR(50), ride\_date DATE

);

INSERT INTO uber\_driver\_rides VALUES (2401, 4201, 'Karthik', 501, 'Ravi', '2025-01-15'),

(2402, 4201, 'Karthik', 501, 'Ravi', '2025-02-10'),

-- (add 8 more for Karthik)

(2411, 4202, 'Deepa', 502, 'Arun', '2025-03-12');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, driver\_id, driver\_name, COUNT(\*) AS ride\_count FROM uber\_driver\_rides

WHERE ride\_date >= DATE\_SUB(CURDATE(), INTERVAL 6 MONTH)

GROUP BY customer\_id, customer\_name, driver\_id, driver\_name HAVING ride\_count >= 10;

## 🩻 Explanation (Tanglish)

* **SELECT customer\_id, customer\_name, driver\_id, driver\_name, COUNT(\*) AS ride\_count:** Customer, driver, and ride count select pannrom.
* **FROM uber\_driver\_rides:** Table select pannrom.
* **WHERE ride\_date >= DATE\_SUB(CURDATE(), INTERVAL 6 MONTH):** Last 6 months data filter pannrom.
* **GROUP BY customer\_id, customer\_name, driver\_id, driver\_name:** Customer and driver wise group pannrom.
* **HAVING ride\_count >= 10:** 10 rides cross pannina users filter pannrom.

**Eg:** Karthik 10 rides Ravi driver kooda pannirukkanga ✅, Deepa Arun driver kooda 1 ride mattum pannirukkanga ❌.

## 🚖 Use Case Value (Tanglish)

Uber and Gowtham can use this to identify **trusted driver-user relationships for premium loyalty plans.**

# 📦 Problem 42: Amazon – Festival Season Buyers

## 🧩 Question (Tanglish)

Gowtham wants to find Amazon customers (like Harini, Ajith) who made purchases during all major festivals in the last year.

## 🗂 Problem Statement (Tanglish)

Amazon-la last 1 year festival season la ellaa major festivals kum purchase pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_festival\_orders ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), festival\_name VARCHAR(50), order\_date DATE

);

INSERT INTO amazon\_festival\_orders VALUES (2501, 4301, 'Harini', 'Diwali', '2024-11-10'),

(2502, 4301, 'Harini', 'Pongal', '2025-01-15'),

(2503, 4301, 'Harini', 'Tamil New Year', '2025-04-14'),

(2504, 4302, 'Ajith', 'Diwali', '2024-11-10');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(DISTINCT festival\_name) AS festivals\_covered

FROM amazon\_festival\_orders

WHERE order\_date >= DATE\_SUB(CURDATE(), INTERVAL 1 YEAR)

GROUP BY customer\_id, customer\_name HAVING festivals\_covered = 3;

## 🩻 Explanation (Tanglish)

* **SELECT customer\_id, customer\_name, COUNT(DISTINCT festival\_name) AS festivals\_covered:** Customer wise unique festivals count pannrom.
* **FROM amazon\_festival\_orders:** Table select pannrom.
* **WHERE order\_date >= DATE\_SUB(CURDATE(), INTERVAL 1 YEAR):** Last 1 year data filter pannrom.
* **GROUP BY customer\_id, customer\_name:** Customer wise group pannrom.
* **HAVING festivals\_covered = 3:** 3 major festivals la purchase pannina customers filter pannrom.

**Eg:** Harini 3 festival purchase pannirukkanga ✅, Ajith 1 festival mattum pannirukkanga ❌.

## 📦 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **festival loyal buyers for festival combo offers.**

# 🎬 Problem 43: Netflix – Documentary Watch Enthusiasts

## 🧩 Question (Tanglish)

Gowtham wants to find Netflix users (like Saran, Divya) who watched at least 5 different documentaries in the last 3 months.

## 🗂 Problem Statement (Tanglish)

Netflix-la last 3 months la 5 different documentaries watch pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE netflix\_documentary\_watch ( watch\_id INT,

user\_id INT,

user\_name VARCHAR(50), documentary\_name VARCHAR(100), watch\_date DATE

);

INSERT INTO netflix\_documentary\_watch VALUES (2601, 4401, 'Saran', 'Planet Earth', '2025-04-05'),

(2602, 4401, 'Saran', 'Our Planet', '2025-04-10'),

(2603, 4401, 'Saran', 'Inside Bill’s Brain', '2025-05-01'),

(2604, 4401, 'Saran', 'The Social Dilemma', '2025-05-15'),

(2605, 4401, 'Saran', 'Explained', '2025-06-01'),

(2606, 4402, 'Divya', 'Planet Earth', '2025-05-10');

## 💡 Solution (MySQL)

SELECT user\_id, user\_name, COUNT(DISTINCT documentary\_name) AS docu\_count FROM netflix\_documentary\_watch

WHERE watch\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

GROUP BY user\_id, user\_name HAVING docu\_count >= 5;

## 🩻 Explanation (Tanglish)

* **SELECT user\_id, user\_name, COUNT(DISTINCT documentary\_name) AS docu\_count:** User wise unique documentaries count pannrom.
* **FROM netflix\_documentary\_watch:** Table select pannrom.

### WHERE watch\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH): Last 3

months data filter pannrom.

* **GROUP BY user\_id, user\_name:** User wise group pannrom.
* **HAVING docu\_count >= 5:** 5 documentaries watch pannina users filter pannrom.

**Eg:** Saran 5 documentaries watch pannirukkanga ✅, Divya 1 documentary mattum pannirukkanga ❌.

## 🎬 Use Case Value (Tanglish)

Netflix and Gowtham can use this to identify **documentary enthusiasts for premium recommendations and niche content targeting.**

# 🚀 Problem 44: Amazon – Same Day Multiple Orders

## 🧩 Question (Tanglish)

Gowtham wants to find Amazon customers (like Surya, Meena) who placed more than 3 orders on the same day in the last 2 months.

## 🗂 Problem Statement (Tanglish)

Amazon-la last 2 months la orae naal la 3 mela orders pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_same\_day\_orders ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), order\_date DATE

);

INSERT INTO amazon\_same\_day\_orders VALUES (2701, 4501, 'Surya', '2025-05-10'),

(2702, 4501, 'Surya', '2025-05-10'),

(2703, 4501, 'Surya', '2025-05-10'),

(2704, 4501, 'Surya', '2025-05-10'),

(2705, 4502, 'Meena', '2025-05-12');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, order\_date, COUNT(\*) AS orders\_count

FROM amazon\_same\_day\_orders

WHERE order\_date >= DATE\_SUB(CURDATE(), INTERVAL 2 MONTH)

GROUP BY customer\_id, customer\_name, order\_date HAVING orders\_count > 3;

## 🩻 Explanation (Tanglish)

### SELECT customer\_id, customer\_name, order\_date, COUNT(\*) AS orders\_count:

Customer, name, date, orders count select pannrom.

* **FROM amazon\_same\_day\_orders:** Table select pannrom.

### WHERE order\_date >= DATE\_SUB(CURDATE(), INTERVAL 2 MONTH): Last 2

months data filter pannrom.

* **GROUP BY customer\_id, customer\_name, order\_date:** Customer and date wise group pannrom.
* **HAVING orders\_count > 3:** Oru naal la 3 mela orders pannina customers filter pannrom.

**Eg:** Surya orae naal la 4 orders pannirukkanga ✅, Meena 1 order mattum pannirukkanga ❌.

## 📦 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **heavy buyers for same-day premium delivery offers.**

# 🎬 Problem 45: Netflix – Users Watching Same Movie Multiple Times

## 🧩 Question (Tanglish)

Gowtham wants to find Netflix users (like Kishore, Aishwarya) who watched the same movie at least 3 times in the last 1 month.

## 🗂 Problem Statement (Tanglish)

Netflix-la last 1 month la orae movie 3 times watch pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE netflix\_repeat\_watch ( watch\_id INT,

user\_id INT,

user\_name VARCHAR(50), movie\_name VARCHAR(100), watch\_date DATE

);

INSERT INTO netflix\_repeat\_watch VALUES (2801, 4601, 'Kishore', 'Inception', '2025-06-01'),

(2802, 4601, 'Kishore', 'Inception', '2025-06-05'),

(2803, 4601, 'Kishore', 'Inception', '2025-06-10'),

(2804, 4602, 'Aishwarya', 'Interstellar', '2025-06-03');

## 💡 Solution (MySQL)

SELECT user\_id, user\_name, movie\_name, COUNT(\*) AS watch\_count FROM netflix\_repeat\_watch

WHERE watch\_date >= DATE\_SUB(CURDATE(), INTERVAL 1 MONTH)

GROUP BY user\_id, user\_name, movie\_name HAVING watch\_count >= 3;

## 🩻 Explanation (Tanglish)

* **SELECT user\_id, user\_name, movie\_name, COUNT(\*) AS watch\_count:** User, name, movie name, watch count select pannrom.
* **FROM netflix\_repeat\_watch:** Table select pannrom.
* **WHERE watch\_date >= DATE\_SUB(CURDATE(), INTERVAL 1 MONTH):** Last 1 month data filter pannrom.
* **GROUP BY user\_id, user\_name, movie\_name:** User and movie wise group pannrom.
* **HAVING watch\_count >= 3:** Oru movie 3 times watch pannina users filter pannrom.

**Eg:** Kishore 'Inception' 3 times watch pannirukkanga ✅, Aishwarya 'Interstellar' 1 time mattum pannirukkanga ❌.

## 🎬 Use Case Value (Tanglish)

Netflix and Gowtham can use this to identify **repeat watchers for movie recommendation refinement.**

# 🚀 Problem 46: Amazon – Wishlist to Purchase Converters

## 🧩 Question (Tanglish)

Gowtham wants to find Amazon customers (like Rahul, Priya) who added products to wishlist and purchased the same within 7 days.

## 🗂 Problem Statement (Tanglish)

Amazon-la wishlist ku add pannina product-ah 7 days kuLLa vaangina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_wishlist ( wishlist\_id INT,

customer\_id INT, customer\_name VARCHAR(50), product\_id INT,

added\_date DATE

);

CREATE TABLE amazon\_purchases ( purchase\_id INT,

customer\_id INT, product\_id INT, purchase\_date DATE

);

INSERT INTO amazon\_wishlist VALUES (2901, 4701, 'Rahul', 601, '2025-06-01'),

(2902, 4702, 'Priya', 602, '2025-06-05');

INSERT INTO amazon\_purchases VALUES (3001, 4701, 601, '2025-06-05'),

(3002, 4702, 602, '2025-06-20');

## 💡 Solution (MySQL)

SELECT w.customer\_id, w.customer\_name, w.product\_id FROM amazon\_wishlist w

JOIN amazon\_purchases p ON w.customer\_id = p.customer\_id AND w.product\_id = p.product\_id

WHERE DATEDIFF(p.purchase\_date, w.added\_date) BETWEEN 0 AND 7;

## 🩻 Explanation (Tanglish)

* **JOIN:** Wishlist and purchases table customer\_id and product\_id moolama join pannrom.
* **DATEDIFF(p.purchase\_date, w.added\_date) BETWEEN 0 AND 7:** Wishlist add pannina date-la irundhu 7 days kuLLa purchase pannirukkangala-nu filter pannrom.

**Eg:** Rahul 4 days kuLLa vaangirukkanga ✅, Priya 15 days la vaangirukkanga ❌.

## 📦 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **wishlist converters for fast-purchase targeting campaigns.**

# 🎬 Problem 47: Netflix – Consistent Weekly Watchers

## 🧩 Question (Tanglish)

Gowtham wants to find Netflix users (like Manoj, Divya) who watched at least 1 movie every week for the last 8 weeks.

## 🗂 Problem Statement (Tanglish)

Netflix-la last 8 weeks la weekly 1 movie watch pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE netflix\_weekly\_watch ( watch\_id INT,

user\_id INT,

user\_name VARCHAR(50), watch\_date DATE

);

INSERT INTO netflix\_weekly\_watch VALUES (3101, 4801, 'Manoj', '2025-04-01'),

(3102, 4801, 'Manoj', '2025-04-08'),

(3103, 4801, 'Manoj', '2025-04-15'),

(3104, 4801, 'Manoj', '2025-04-22'),

(3105, 4801, 'Manoj', '2025-04-29'),

(3106, 4801, 'Manoj', '2025-05-06'),

(3107, 4801, 'Manoj', '2025-05-13'),

(3108, 4801, 'Manoj', '2025-05-20'),

(3109, 4802, 'Divya', '2025-05-10');

## 💡 Solution (MySQL)

SELECT user\_id, user\_name, COUNT(DISTINCT YEARWEEK(watch\_date)) AS weeks\_watched

FROM netflix\_weekly\_watch

WHERE watch\_date >= DATE\_SUB(CURDATE(), INTERVAL 8 WEEK)

GROUP BY user\_id, user\_name HAVING weeks\_watched = 8;

## 🩻 Explanation (Tanglish)

* **YEARWEEK(watch\_date):** Watch week identify panna use pannrom.
* **COUNT(DISTINCT YEARWEEK(watch\_date)) = 8:** 8 weeks la continuous watch pannina users filter pannrom.

**Eg:** Manoj 8 weeks la watch pannirukkanga ✅, Divya oru week mattum pannirukkanga ❌.

## 🎬 Use Case Value (Tanglish)

Netflix and Gowtham can use this to identify **weekly consistent viewers for retention plans.**

# 🚖 Problem 48: Uber – Peak Hour Riders

## 🧩 Question (Tanglish)

Gowtham wants to find Uber customers (like Hari, Sangeetha) who took rides during peak hours (8 AM - 10 AM) at least 10 times in the last 2 months.

## 🗂 Problem Statement (Tanglish)

Uber-la last 2 months la peak hours (8-10 AM) la 10 rides pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE uber\_peak\_rides ( ride\_id INT,

customer\_id INT, customer\_name VARCHAR(50), ride\_time TIME,

ride\_date DATE

);

INSERT INTO uber\_peak\_rides VALUES (3201, 4901, 'Hari', '08:15:00', '2025-05-01'),

(3202, 4901, 'Hari', '09:00:00', '2025-05-03'),

-- (add 8 more for Hari)

(3211, 4902, 'Sangeetha', '07:45:00', '2025-05-05');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS peak\_rides FROM uber\_peak\_rides

WHERE TIME(ride\_time) BETWEEN '08:00:00' AND '10:00:00' AND ride\_date >= DATE\_SUB(CURDATE(), INTERVAL 2 MONTH)

GROUP BY customer\_id, customer\_name HAVING peak\_rides >= 10;

## 🩻 Explanation (Tanglish)

* **TIME(ride\_time) BETWEEN '08:00:00' AND '10:00:00':** 8-10 AM rides filter pannrom.
* **COUNT(\*) >= 10:** 10 rides cross pannina customers filter pannrom.

**Eg:** Hari 10 rides pannirukkanga ✅, Sangeetha 7:45 la irundhudhu ❌.

## 🚖 Use Case Value (Tanglish)

Uber and Gowtham can use this to identify **peak hour users for surge pricing and targeted discounts.**

# 📦 Problem 49: Amazon – Subscription Box Repeat Buyers

## 🧩 Question (Tanglish)

Gowtham wants to find Amazon customers (like Arjun, Sneha) who purchased the same subscription box for 3 consecutive months.

## 🗂 Problem Statement (Tanglish)

Amazon-la orae subscription box 3 months continuous-a vaangina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_subscription\_orders ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), subscription\_box VARCHAR(50), order\_date DATE

);

INSERT INTO amazon\_subscription\_orders VALUES (3301, 5001, 'Arjun', 'Wellness Box', '2025-03-05'),

(3302, 5001, 'Arjun', 'Wellness Box', '2025-04-07'),

(3303, 5001, 'Arjun', 'Wellness Box', '2025-05-06'),

(3304, 5002, 'Sneha', 'Grooming Box', '2025-05-10');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, subscription\_box, COUNT(DISTINCT DATE\_FORMAT(order\_date, '%Y-%m')) AS months\_count

FROM amazon\_subscription\_orders

WHERE order\_date >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

GROUP BY customer\_id, customer\_name, subscription\_box HAVING months\_count = 3;

## 🩻 Explanation (Tanglish)

* **DATE\_FORMAT(order\_date, '%Y-%m'):** Month wise group panna format pannrom.
* **COUNT(DISTINCT month) = 3:** 3 months continuous subscription buyers filter pannrom.

**Eg:** Arjun 3 months continuous purchase pannirukkanga ✅, Sneha 1 month mattum pannirukkanga ❌.

## 📦 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **subscription loyal buyers for rewards.**

# 🎬 Problem 50: Netflix – Genre Consistency Watchers

## 🧩 Question (Tanglish)

Gowtham wants to find Netflix users (like Kavin, Asha) who watched movies of the same genre for 4 consecutive weeks.

## 🗂 Problem Statement (Tanglish)

Netflix-la 4 weeks continuous-a orae genre movies watch pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE netflix\_genre\_consistency ( watch\_id INT,

user\_id INT,

user\_name VARCHAR(50), genre VARCHAR(50),

watch\_date DATE

);

INSERT INTO netflix\_genre\_consistency VALUES (3401, 5101, 'Kavin', 'Thriller', '2025-05-01'),

(3402, 5101, 'Kavin', 'Thriller', '2025-05-08'),

(3403, 5101, 'Kavin', 'Thriller', '2025-05-15'),

(3404, 5101, 'Kavin', 'Thriller', '2025-05-22'),

(3405, 5102, 'Asha', 'Drama', '2025-05-10');

## 💡 Solution (MySQL)

SELECT user\_id, user\_name, genre, COUNT(DISTINCT YEARWEEK(watch\_date)) AS weeks\_count

FROM netflix\_genre\_consistency

WHERE watch\_date >= DATE\_SUB(CURDATE(), INTERVAL 1 MONTH)

GROUP BY user\_id, user\_name, genre HAVING weeks\_count = 4;

## 🩻 Explanation (Tanglish)

* **YEARWEEK(watch\_date):** Week wise identify panna.
* **COUNT DISTINCT week = 4:** 4 weeks continuous same genre watch pannina users filter pannrom.

**Eg:** Kavin 4 weeks continuous Thriller watch pannirukkanga ✅, Asha 1 week mattum pannirukkanga ❌.

## 🎬 Use Case Value (Tanglish)

Netflix and Gowtham can use this to identify **genre loyal viewers for niche content promotion.**

# 🏏 Problem 51: Amazon – Orders Placed on Every Cricket Match Day

## 🧩 Question (Tanglish)

Gowtham wants to find Amazon customers (like Arun, Divya) who placed at least 1 order on

**every IPL cricket match day** in the last season.

## 🗂 Problem Statement (Tanglish)

IPL cricket match days la ellathulayum atleast 1 order pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE cricket\_match\_days ( match\_date DATE

);

CREATE TABLE amazon\_orders\_cricket ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), order\_date DATE

);

INSERT INTO cricket\_match\_days VALUES ('2025-04-01'), ('2025-04-05'), ('2025-04-10');

INSERT INTO amazon\_orders\_cricket VALUES (401, 6001, 'Arun', '2025-04-01'),

(402, 6001, 'Arun', '2025-04-05'),

(403, 6001, 'Arun', '2025-04-10'),

(404, 6002, 'Divya', '2025-04-01'),

(405, 6002, 'Divya', '2025-04-05');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name

FROM amazon\_orders\_cricket

WHERE order\_date IN (SELECT match\_date FROM cricket\_match\_days) GROUP BY customer\_id, customer\_name

HAVING COUNT(DISTINCT order\_date) = (SELECT COUNT(\*) FROM cricket\_match\_days);

## 🩻 Explanation (Tanglish)

* **cricket\_match\_days:** IPL match dates maintain pannum table.
* **amazon\_orders\_cricket:** Orders table with customer and order date.
* **WHERE order\_date IN (match\_date):** Match days la irukkara orders filter pannrom.
* **GROUP BY customer\_id, customer\_name:** Customer wise group pannrom.
* **HAVING COUNT = total match days:** Ellaa match days la order pannina customers filter pannrom.

**Eg:** Arun ellaa 3 match days la order pannirukkanga ✅, Divya 2 match days la mattum pannirukkanga ❌.

## 🏏 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **cricket season loyal buyers for targeted sports campaign offers.**

# 🏏 Problem 52: Swiggy – Orders Placed During Every Rainy Day

## 🧩 Question (Tanglish)

Gowtham wants to find Swiggy customers (like Bala, Priya) who placed at least 1 order on

**every rainy day** in Chennai last month.

## 🗂 Problem Statement (Tanglish)

Chennai last month rainy days la ellathulayum atleast 1 order pannina Swiggy customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE chennai\_rainy\_days ( rainy\_date DATE

);

CREATE TABLE swiggy\_orders\_rain ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), order\_date DATE

);

INSERT INTO chennai\_rainy\_days VALUES ('2025-06-01'), ('2025-06-05'), ('2025-06-10');

INSERT INTO swiggy\_orders\_rain VALUES (501, 6101, 'Bala', '2025-06-01'),

(502, 6101, 'Bala', '2025-06-05'),

(503, 6101, 'Bala', '2025-06-10'),

(504, 6102, 'Priya', '2025-06-01'),

(505, 6102, 'Priya', '2025-06-05');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name FROM swiggy\_orders\_rain

WHERE order\_date IN (SELECT rainy\_date FROM chennai\_rainy\_days) GROUP BY customer\_id, customer\_name

HAVING COUNT(DISTINCT order\_date) = (SELECT COUNT(\*) FROM chennai\_rainy\_days);

## 🩻 Explanation (Tanglish)

* **chennai\_rainy\_days:** Chennai rainy days maintain panna table.
* **swiggy\_orders\_rain:** Swiggy orders with customer and order date.
* **WHERE order\_date IN (rainy\_date):** Rainy days la irukkara orders filter pannrom.
* **GROUP BY customer\_id, customer\_name:** Customer wise group pannrom.
* **HAVING COUNT = total rainy days:** Ellaa rainy days la order pannina customers filter pannrom.

**Eg:** Bala ellaa 3 rainy days la order pannirukkanga ✅, Priya 2 days la mattum pannirukkanga

❌.

## ☔ Use Case Value (Tanglish)

Swiggy and Gowtham can use this to identify **rainy day loyal customers for weather-based offers.**

# 🎬 Problem 53: Netflix – Users Watching New Releases on Release Day

## 🧩 Question (Tanglish)

Gowtham wants to find Netflix users (like Karthik, Anjali) who watched every **new Tamil movie release on the release day** in the last 2 months.

## 🗂 Problem Statement (Tanglish)

Netflix-la last 2 months la Tamil new release movies release day la ellathayum watch pannina users identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE tamil\_movie\_releases ( movie\_name VARCHAR(100), release\_date DATE

);

CREATE TABLE netflix\_watch\_release\_day ( watch\_id INT,

user\_id INT,

user\_name VARCHAR(50), movie\_name VARCHAR(100), watch\_date DATE

);

INSERT INTO tamil\_movie\_releases VALUES ('Movie A', '2025-05-01'),

('Movie B', '2025-05-10'),

('Movie C', '2025-05-20');

INSERT INTO netflix\_watch\_release\_day VALUES (601, 6201, 'Karthik', 'Movie A', '2025-05-01'),

(602, 6201, 'Karthik', 'Movie B', '2025-05-10'),

(603, 6201, 'Karthik', 'Movie C', '2025-05-20'),

(604, 6202, 'Anjali', 'Movie A', '2025-05-01'),

(605, 6202, 'Anjali', 'Movie B', '2025-05-10');

💡 **Solution (MySQL)** SELECT user\_id, user\_name FROM netflix\_watch\_release\_day

WHERE (movie\_name, watch\_date) IN (SELECT movie\_name, release\_date FROM tamil\_movie\_releases)

GROUP BY user\_id, user\_name

HAVING COUNT(DISTINCT movie\_name) = (SELECT COUNT(\*) FROM

tamil\_movie\_releases);

## 🩻 Explanation (Tanglish)

* **tamil\_movie\_releases:** Tamil movie releases with name and release date maintain pannrom.
* **netflix\_watch\_release\_day:** User watch data with movie and watch date maintain pannrom.
* **WHERE (movie\_name, watch\_date) IN (movie\_name, release\_date):** Release day la irukkara watch records filter pannrom.
* **GROUP BY user\_id, user\_name:** User wise group pannrom.
* **HAVING COUNT = total releases:** Ellaa releases day la watch pannina users filter pannrom.

**Eg:** Karthik 3 releases day la watch pannirukkanga ✅, Anjali 2 releases la mattum pannirukkanga ❌.

## 🎬 Use Case Value (Tanglish)

Netflix and Gowtham can use this to identify **release day loyal viewers for early access and premium content campaigns.**

# 🍬 Problem 54: Amazon – Customers Who Only Ordered Sweets During Festivals

## 🧩 Question (Tanglish)

Find customers who **only ordered sweets** during festivals.

## 🗂 Problem Statement (Tanglish)

Festival days la sweets mattum than order pannina Amazon customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE festival\_days ( festival\_date DATE

);

CREATE TABLE amazon\_orders\_festival ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), product\_category VARCHAR(50), order\_date DATE

);

INSERT INTO festival\_days VALUES

('2025-10-25'), ('2025-11-10'), ('2025-12-05');

INSERT INTO amazon\_orders\_festival VALUES (701, 6301, 'Suresh', 'Sweets', '2025-10-25'),

(702, 6301, 'Suresh', 'Sweets', '2025-11-10'),

(703, 6301, 'Sweets', '2025-12-05'),

(704, 6302, 'Kavitha', 'Sweets', '2025-10-25'),

(705, 6302, 'Kavitha', 'Snacks', '2025-11-10');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name FROM amazon\_orders\_festival

WHERE order\_date IN (SELECT festival\_date FROM festival\_days) GROUP BY customer\_id, customer\_name

HAVING SUM(product\_category != 'Sweets') = 0;

## 🩻 Explanation (Tanglish)

* **festival\_days:** Festival dates maintain panna table.
* **amazon\_orders\_festival:** Customer order data with category maintain pannrom.
* **WHERE order\_date IN (festival\_date):** Festival days la orders filter pannrom.
* **GROUP BY customer\_id, customer\_name:** Customer wise group pannrom.
* **HAVING SUM(product\_category != 'Sweets') = 0:** Festival days la sweets mattum than order pannina customers filter pannrom.

**Eg:** Suresh ellaa festival days la sweets mattum order pannirukkanga ✅, Kavitha oru festival la snacks order pannirukkanga ❌.

## 🍬 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **sweet loyalists for festival targeted sweet box campaigns.**

# 🚨 Problem 55: Swiggy – Orders Placed During Curfew Periods

## 🧩 Question (Tanglish)

Gowtham wants to find customers who ordered during **curfew periods.**

## 🗂 Problem Statement (Tanglish)

Curfew time la order pannina Swiggy customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE curfew\_periods ( curfew\_date DATE

);

CREATE TABLE swiggy\_orders\_curfew ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), order\_date DATE

);

INSERT INTO curfew\_periods VALUES ('2025-05-01'), ('2025-05-02'), ('2025-05-03');

INSERT INTO swiggy\_orders\_curfew VALUES (801, 6401, 'Hari', '2025-05-01'),

(802, 6401, 'Hari', '2025-05-02'),

(803, 6401, 'Hari', '2025-05-03'),

(804, 6402, 'Anu', '2025-05-01');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name FROM swiggy\_orders\_curfew

WHERE order\_date IN (SELECT curfew\_date FROM curfew\_periods) GROUP BY customer\_id, customer\_name

HAVING COUNT(DISTINCT order\_date) = (SELECT COUNT(\*) FROM curfew\_periods);

## 🩻 Explanation (Tanglish)

* **curfew\_periods:** Curfew dates maintain panna table.
* **swiggy\_orders\_curfew:** Orders data maintain pannrom.
* **WHERE order\_date IN (curfew\_date):** Curfew days orders filter pannrom.
* **GROUP BY customer\_id, customer\_name:** Customer wise group pannrom.
* **HAVING COUNT = total curfew days:** Ellaa curfew days la order pannina customers filter pannrom.

**Eg:** Hari ellaa curfew days la order pannirukkanga ✅, Anu oru naal mattum pannirukkanga ❌.

## 🚨 Use Case Value (Tanglish)

Swiggy and Gowtham can use this to identify **essential demand customers during curfew for prioritization.**

# 💊 Problem 56: Swiggy – Customers Who Ordered Medicine and Food on Same Day

## 🧩 Question (Tanglish)

Identify customers who **ordered medicine and food on the same day.**

## 🗂 Problem Statement (Tanglish)

Oru naal la medicine-um food-um order pannina Swiggy customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE swiggy\_orders\_same\_day ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), category VARCHAR(20), order\_date DATE

);

INSERT INTO swiggy\_orders\_same\_day VALUES (901, 6501, 'Manoj', 'Food', '2025-06-10'),

(902, 6501, 'Manoj', 'Medicine', '2025-06-10'),

(903, 6502, 'Revathi', 'Food', '2025-06-11');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, order\_date

FROM swiggy\_orders\_same\_day WHERE category IN ('Food', 'Medicine')

GROUP BY customer\_id, customer\_name, order\_date HAVING COUNT(DISTINCT category) = 2;

## 🩻 Explanation (Tanglish)

* **category IN ('Food', 'Medicine'):** Food and Medicine orders filter pannrom.
* **GROUP BY customer\_id, customer\_name, order\_date:** Customer and date wise group pannrom.
* **HAVING COUNT = 2:** Andha naal la rendu category um order pannina customers filter pannrom.

**Eg:** Manoj andha naal la food and medicine order pannirukkanga ✅, Revathi food mattum pannirukkanga ❌.

## 💊 Use Case Value (Tanglish)

Swiggy and Gowtham can use this to identify **health-conscious multi-category users for bundle offers.**

# 🚫 Problem 57: Swiggy – Customers Who Cancelled Orders Due to Late Delivery

## 🧩 Question (Tanglish)

Find customers who cancelled orders due to **late delivery.**

## 🗂 Problem Statement (Tanglish)

Late delivery naala order cancel pannina Swiggy customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE swiggy\_order\_cancellations ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50),

cancellation\_reason VARCHAR(50)

);

INSERT INTO swiggy\_order\_cancellations VALUES (1001, 6601, 'Rahul', 'Late Delivery'),

(1002, 6601, 'Rahul', 'Late Delivery'),

(1003, 6602, 'Lakshmi', 'Other');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS late\_cancellations FROM swiggy\_order\_cancellations

WHERE cancellation\_reason = 'Late Delivery' GROUP BY customer\_id, customer\_name;

## 🩻 Explanation (Tanglish)

* **cancellation\_reason = 'Late Delivery':** Late delivery reason la cancel pannina orders filter pannrom.
* **GROUP BY customer\_id, customer\_name:** Customer wise group pannrom.
* **COUNT:** Late delivery cancellations count kaanom.

**Eg:** Rahul 2 late delivery cancellations pannirukkanga ✅, Lakshmi different reason kaaga pannirukkanga ❌.

## 🚫 Use Case Value (Tanglish)

Swiggy and Gowtham can use this to identify **logistics issues for customer compensation and delivery improvements.**

# 💰 Problem 58: Amazon – Customers Who Placed Only Prepaid Orders

## 🧩 Question (Tanglish)

Identify customers who **placed only prepaid orders.**

## 🗂 Problem Statement (Tanglish)

Prepaid orders mattum than pottu irukkara Amazon customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_orders\_payment ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), payment\_type VARCHAR(20)

);

INSERT INTO amazon\_orders\_payment VALUES (1101, 6701, 'Sathish', 'Prepaid'),

(1102, 6701, 'Sathish', 'Prepaid'),

(1103, 6702, 'Meera', 'Prepaid'),

(1104, 6702, 'Meera', 'COD');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name FROM amazon\_orders\_payment

GROUP BY customer\_id, customer\_name HAVING SUM(payment\_type != 'Prepaid') = 0;

## 🩻 Explanation (Tanglish)

* **payment\_type != 'Prepaid':** COD or other payment types check pannrom.
* **SUM = 0:** Prepaid mattum than irukka nu verify pannrom.

**Eg:** Sathish prepaid orders mattum pannirukkanga ✅, Meera COD um pottirukkanga ❌.

## 💰 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **prepaid loyal customers for reward offers and fast processing.**

# 🍽 Problem 59: Swiggy – Customers Who Ordered from High-Rated Restaurants Only

## 🧩 Question (Tanglish)

Find customers who **ordered from high-rated restaurants only.**

## 🗂 Problem Statement (Tanglish)

High-rated restaurants la mattum order pannina Swiggy customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE swiggy\_restaurant\_ratings ( restaurant\_id INT,

restaurant\_name VARCHAR(50), rating DECIMAL(2,1)

);

CREATE TABLE swiggy\_orders\_restaurants ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), restaurant\_id INT

);

INSERT INTO swiggy\_restaurant\_ratings VALUES (201, 'Annapoorna', 4.5),

(202, 'QuickBites', 3.8);

INSERT INTO swiggy\_orders\_restaurants VALUES (1201, 6801, 'Ravi', 201),

(1202, 6801, 'Ravi', 201),

(1203, 6802, 'Divya', 202);

## 💡 Solution (MySQL)

SELECT o.customer\_id, o.customer\_name FROM swiggy\_orders\_restaurants o

JOIN swiggy\_restaurant\_ratings r ON o.restaurant\_id = r.restaurant\_id GROUP BY o.customer\_id, o.customer\_name

HAVING SUM(r.rating < 4.0) = 0;

## 🩻 Explanation (Tanglish)

* **JOIN:** Orders table and restaurant ratings table join pannrom.
* **r.rating < 4.0:** 4 rating kuda keela irundha restaurants eduththu filter pannrom.
* **SUM = 0:** High-rated restaurants la mattum than irukka nu check pannrom.

**Eg:** Ravi high-rated restaurant la mattum order pannirukkanga ✅, Divya low-rated la order pannirukkanga ❌.

## 🍽 Use Case Value (Tanglish)

Swiggy and Gowtham can use this to identify **premium food customers for exclusive premium offers.**

# 🎉 Problem 60: Swiggy – Customers Who Placed Bulk Orders for Parties

## 🧩 Question (Tanglish)

Gowtham wants to find customers who placed **bulk orders for parties.**

## 🗂 Problem Statement (Tanglish)

Swiggy-la party ku bulk orders pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE swiggy\_bulk\_party\_orders ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), quantity INT,

order\_date DATE

);

INSERT INTO swiggy\_bulk\_party\_orders VALUES (1301, 6901, 'Vikram', 25, '2025-06-10'),

(1302, 6901, 'Vikram', 30, '2025-06-15'),

(1303, 6902, 'Keerthi', 8, '2025-06-12');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS party\_bulk\_orders FROM swiggy\_bulk\_party\_orders

WHERE quantity >= 20

GROUP BY customer\_id, customer\_name;

## 🩻 Explanation (Tanglish)

* **quantity >= 20:** 20 items mela order pannina records filter pannrom.
* **GROUP BY customer\_id, customer\_name:** Customer wise group pannrom.
* **COUNT:** Party bulk orders count kaanom.

**Eg:** Vikram 2 bulk orders pannirukkanga ✅, Keerthi quantity kammiya irukku ❌.

## 🎉 Use Case Value (Tanglish)

Swiggy and Gowtham can use this to identify **party order customers for special catering offers.**

# ✅ Problem 61: Amazon – Customers Who Ordered Electronics During Sale Days

## 🧩 Question (Tanglish)

Find customers who ordered **electronics during sale days.**

## 🗂 Problem Statement (Tanglish)

Sale days la electronics order pannina Amazon customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE sale\_days ( sale\_date DATE

);

CREATE TABLE amazon\_orders\_sale ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), category VARCHAR(50), order\_date DATE

);

INSERT INTO sale\_days VALUES ('2025-07-01'), ('2025-07-05');

INSERT INTO amazon\_orders\_sale VALUES (1401, 7001, 'Aravind', 'Electronics', '2025-07-01'),

(1402, 7002, 'Deepa', 'Clothing', '2025-07-01'),

(1403, 7001, 'Aravind', 'Electronics', '2025-07-05');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS electronics\_sale\_orders FROM amazon\_orders\_sale

WHERE category = 'Electronics' AND order\_date IN (SELECT sale\_date FROM sale\_days) GROUP BY customer\_id, customer\_name;

## 🩻 Explanation (Tanglish)

* **category = 'Electronics':** Electronics orders mattum filter pannrom.
* **order\_date IN (sale\_date):** Sale days la irukkara orders filter pannrom.
* **GROUP BY customer\_id, customer\_name:** Customer wise group pannrom.

**Eg:** Aravind 2 sale day electronics orders pannirukkanga ✅, Deepa electronics vaangala ❌.

## ✅ Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **electronics sale day buyers for exclusive flash deals.**

# 🎶 Problem 62: Netflix – Customers Who Watched Music Concerts

## 🧩 Question (Tanglish)

Find customers who watched **music concerts on Netflix.**

## 🗂 Problem Statement (Tanglish)

Netflix-la music concert content watch pannina users identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE netflix\_concert\_watch ( watch\_id INT,

user\_id INT,

user\_name VARCHAR(50), content\_type VARCHAR(50), watch\_date DATE

);

INSERT INTO netflix\_concert\_watch VALUES (1501, 7101, 'Senthil', 'Music Concert', '2025-06-01'),

(1502, 7102, 'Anitha', 'Movie', '2025-06-02'),

(1503, 7101, 'Senthil', 'Music Concert', '2025-06-05');

## 💡 Solution (MySQL)

SELECT user\_id, user\_name, COUNT(\*) AS concert\_views FROM netflix\_concert\_watch

WHERE content\_type = 'Music Concert' GROUP BY user\_id, user\_name;

## 🩻 Explanation (Tanglish)

* **content\_type = 'Music Concert':** Music concert content watch pannina records filter pannrom.
* **GROUP BY user\_id, user\_name:** User wise group pannrom.
* **COUNT:** Concert views count kaanom.

**Eg:** Senthil 2 music concerts watch pannirukkanga ✅, Anitha movie mattum watch pannirukkanga ❌.

## 🎶 Use Case Value (Tanglish)

Netflix and Gowtham can use this to identify **music content lovers for concert recommendations.**

# 🛍 Problem 63: Amazon – Customers Who Purchased Only Branded Products for a Year

## 🧩 Question (Tanglish)

Find customers who purchased **only branded products for a year.**

## 🗂 Problem Statement (Tanglish)

1 year kuLLa branded products mattum than purchase pannina Amazon customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_orders\_brand ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), is\_branded BOOLEAN, order\_date DATE

);

INSERT INTO amazon\_orders\_brand VALUES (1601, 7201, 'Gowtham', TRUE, '2024-09-10'),

(1602, 7201, 'Gowtham', TRUE, '2025-01-15'),

(1603, 7202, 'Lavanya', TRUE, '2025-03-20'),

(1604, 7202, 'Lavanya', FALSE, '2025-04-18');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name FROM amazon\_orders\_brand

WHERE order\_date >= DATE\_SUB(CURDATE(), INTERVAL 1 YEAR)

GROUP BY customer\_id, customer\_name HAVING SUM(is\_branded = FALSE) = 0;

## 🩻 Explanation (Tanglish)

* **order\_date >= last 1 year:** Last 1 year data filter pannrom.
* **is\_branded = FALSE:** Non-branded products check pannrom.
* **SUM = 0:** Non-branded purchase pannatha customers filter pannrom.

**Eg:** Gowtham branded products mattum purchase pannirukkanga ✅, Lavanya branded illa oru product purchase pannirukkanga ❌.

## 🛍 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **branded product loyal customers for premium campaign offers.**

# 🎁 Problem 64: Amazon – Customers Who Did Gift Purchases Frequently

## 🧩 Question (Tanglish)

Identify customers who did **gift purchases frequently**.

## 🗂 Problem Statement (Tanglish)

Gift purchases 5 times mela pannina Amazon customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_gift\_orders ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), is\_gift BOOLEAN,

order\_date DATE

);

INSERT INTO amazon\_gift\_orders VALUES (1701, 7301, 'Gowtham', TRUE, '2025-02-10'),

(1702, 7301, 'Gowtham', TRUE, '2025-03-12'),

(1703, 7301, 'Gowtham', TRUE, '2025-04-15'),

(1704, 7301, 'Gowtham', TRUE, '2025-05-18'),

(1705, 7301, 'Gowtham', TRUE, '2025-06-20'),

(1706, 7301, 'Gowtham', TRUE, '2025-07-01'),

(1707, 7302, 'Ravi', TRUE, '2025-06-10');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS gift\_count FROM amazon\_gift\_orders

WHERE is\_gift = TRUE

GROUP BY customer\_id, customer\_name HAVING gift\_count > 5;

## 🩻 Explanation (Tanglish)

* **is\_gift = TRUE:** Gift ah order panninathu filter pannrom.
* **COUNT > 5:** 5 times mela gift purchase pannina customers filter pannrom.

**Eg:** Gowtham 6 times gift order pannirukkanga ✅, Ravi 1 time mattum pannirukkanga ❌.

## 🎁 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **gift loyal customers for festival and birthday campaigns.**

# ⚡ Problem 65: Amazon – Customers Who Purchased During Flash Sales

## 🧩 Question (Tanglish)

Gowtham wants to find customers who purchased during **flash sales**.

## 🗂 Problem Statement (Tanglish)

Flash sales time la purchase pannina Amazon customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE flash\_sale\_days ( sale\_date DATE

);

CREATE TABLE amazon\_flash\_orders ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), order\_date DATE

);

INSERT INTO flash\_sale\_days VALUES ('2025-07-05'), ('2025-07-15');

INSERT INTO amazon\_flash\_orders VALUES (1801, 7401, 'Karthik', '2025-07-05'),

(1802, 7401, 'Karthik', '2025-07-15'),

(1803, 7402, 'Sahana', '2025-07-05');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS flash\_sale\_orders FROM amazon\_flash\_orders

WHERE order\_date IN (SELECT sale\_date FROM flash\_sale\_days) GROUP BY customer\_id, customer\_name;

## 🩻 Explanation (Tanglish)

* **order\_date IN sale days:** Flash sale days la orders filter pannrom.
* **COUNT:** Flash sale la eduththa orders count kaanom.

**Eg:** Karthik 2 flash sale orders pannirukkanga ✅, Sahana 1 order pannirukkanga ✅.

## ⚡ Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **flash sale participants for exclusive early bird offers.**

# 💸 Problem 66: Amazon – Customers Who Bought Products Only in Offers

## 🧩 Question (Tanglish)

Identify customers who bought products **only in offers.**

## 🗂 Problem Statement (Tanglish)

Offer time la mattum than purchase pannina Amazon customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_offer\_orders ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), is\_offer BOOLEAN,

order\_date DATE

);

INSERT INTO amazon\_offer\_orders VALUES (1901, 7501, 'Gowtham', TRUE, '2025-03-15'),

(1902, 7501, 'Gowtham', TRUE, '2025-04-20'),

(1903, 7502, 'Divya', TRUE, '2025-05-10'),

(1904, 7502, 'Divya', FALSE, '2025-06-15');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name FROM amazon\_offer\_orders

GROUP BY customer\_id, customer\_name HAVING SUM(is\_offer = FALSE) = 0;

## 🩻 Explanation (Tanglish)

* **is\_offer = FALSE:** Offer illa purchases irukka check pannrom.
* **SUM = 0:** Offer illa purchase pannatha customers filter pannrom.

**Eg:** Gowtham offer time la mattum purchase pannirukkanga ✅, Divya offer illa oru order pannirukkanga ❌.

## 💸 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **deal hunters for targeted deal notifications.**

# 🌙 Problem 67: Amazon – Customers Who Purchased During Midnight Offers

## 🧩 Question (Tanglish)

Find customers who purchased during **midnight offers**.

## 🗂 Problem Statement (Tanglish)

Midnight (12 AM - 2 AM) offer la purchase pannina Amazon customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_midnight\_orders ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), order\_time TIME,

order\_date DATE

);

INSERT INTO amazon\_midnight\_orders VALUES (2001, 7601, 'Gowtham', '00:15:00', '2025-07-20'),

(2002, 7601, 'Gowtham', '01:30:00', '2025-07-22'),

(2003, 7602, 'Meera', '23:50:00', '2025-07-21');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS midnight\_orders FROM amazon\_midnight\_orders

WHERE TIME(order\_time) BETWEEN '00:00:00' AND '02:00:00'

GROUP BY customer\_id, customer\_name;

## 🩻 Explanation (Tanglish)

* **TIME(order\_time) BETWEEN 12 AM - 2 AM:** Midnight orders filter pannrom.
* **COUNT:** Midnight orders count kaanom.

**Eg:** Gowtham 2 midnight orders pannirukkanga ✅, Meera night la pannirukkanga aana midnight illa ❌.

## 🌙 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **night shoppers for midnight flash sale campaigns.**

# 📱 Problem 68: Amazon – Customers Who Purchased Mobile and Accessories Together

## 🧩 Question (Tanglish)

Gowtham wants to identify customers who purchased **mobile and accessories together**.

## 🗂 Problem Statement (Tanglish)

Oru naal la mobile-um accessories-um purchase pannina Amazon customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_mobile\_accessories\_orders ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), category VARCHAR(50), order\_date DATE

);

INSERT INTO amazon\_mobile\_accessories\_orders VALUES (2101, 7701, 'Karthik', 'Mobile', '2025-06-15'),

(2102, 7701, 'Karthik', 'Accessories', '2025-06-15'),

(2103, 7702, 'Sneha', 'Mobile', '2025-06-16');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, order\_date FROM amazon\_mobile\_accessories\_orders WHERE category IN ('Mobile', 'Accessories')

GROUP BY customer\_id, customer\_name, order\_date HAVING COUNT(DISTINCT category) = 2;

## 🩻 Explanation (Tanglish)

* **category IN ('Mobile','Accessories'):** Mobile and accessories categories filter pannrom.
* **GROUP BY customer\_id, customer\_name, order\_date:** Customer and date wise group pannrom.
* **HAVING COUNT = 2:** Andha naal la rendu category um purchase pannina customers filter pannrom.

**Eg:** Karthik same day la mobile and accessories vaangirukkanga ✅, Sneha mobile mattum vaangirukkanga ❌.

## 📱 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **bundle buyers for combo offers.**

# 🔄 Problem 69: Amazon – Customers Who Placed Return Requests Immediately After Delivery

## 🧩 Question (Tanglish)

Identify customers who placed **return requests immediately after delivery**.

## 🗂 Problem Statement (Tanglish)

Delivery aana adutha naal immediate-a return request potta customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_returns (

return\_id INT, customer\_id INT,

customer\_name VARCHAR(50), delivery\_date DATE, return\_request\_date DATE

);

INSERT INTO amazon\_returns VALUES

(2201, 7801, 'Arjun', '2025-07-01', '2025-07-02'),

(2202, 7802, 'Divya', '2025-07-01', '2025-07-05');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS immediate\_returns FROM amazon\_returns

WHERE DATEDIFF(return\_request\_date, delivery\_date) = 1 GROUP BY customer\_id, customer\_name;

## 🩻 Explanation (Tanglish)

* **DATEDIFF = 1:** Delivery date ku adutha naal immediate-a return request potta customers filter pannrom.
* **COUNT:** Immediate returns count kaanom.

**Eg:** Arjun immediate return pannirukkanga ✅, Divya 4 naal ku apram return request pannirukkanga ❌.

## 🔄 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **frequent immediate return customers for quality analysis and fraud checks.**

# 💳 Problem 70: Amazon – Customers Who Used EMI for All Purchases

## 🧩 Question (Tanglish)

Gowtham wants to identify customers who used **EMI for all their purchases**.

## 🗂 Problem Statement (Tanglish)

EMI option use pannitu mattum than purchase pannina Amazon customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE amazon\_emi\_orders ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), payment\_mode VARCHAR(20), order\_date DATE

);

INSERT INTO amazon\_emi\_orders VALUES (2301, 7901, 'Gowtham', 'EMI', '2025-02-01'),

(2302, 7901, 'Gowtham', 'EMI', '2025-04-15'),

(2303, 7902, 'Nisha', 'COD', '2025-03-20');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name FROM amazon\_emi\_orders

GROUP BY customer\_id, customer\_name HAVING SUM(payment\_mode != 'EMI') = 0;

## 🩻 Explanation (Tanglish)

* **payment\_mode != 'EMI':** EMI illa purchases check pannrom.
* **SUM = 0:** EMI illa purchase pannatha customers filter pannrom.

**Eg:** Gowtham EMI la mattum purchase pannirukkanga ✅, Nisha COD use pannirukkanga ❌.

## 💳 Use Case Value (Tanglish)

Amazon and Gowtham can use this to identify **EMI dependent customers for no-cost EMI promotions and upsell campaigns.**

# Problem 71: Identify Users Who Created Multiple Accounts Using the Same IP

## 🧩 Question (Tanglish)

Gowtham wants to find users who created **multiple accounts using the same IP address.**

## 🗂 Problem Statement (Tanglish)

Oru IP address la multiple accounts create pannina users identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE user\_accounts ( user\_id INT,

user\_name VARCHAR(50), ip\_address VARCHAR(50), created\_date DATE

);

INSERT INTO user\_accounts VALUES

(1, 'Gowtham', '192.168.0.1', '2025-01-01'),

(2, 'Ravi', '192.168.0.1', '2025-01-02'),

(3, 'Priya', '192.168.0.2', '2025-01-03');

## 💡 Solution (MySQL)

SELECT ip\_address, COUNT(\*) AS account\_count FROM user\_accounts

GROUP BY ip\_address HAVING account\_count > 1;

## 🩻 Explanation (Tanglish)

* **GROUP BY ip\_address:** IP address wise group pannrom.
* **COUNT > 1:** Oru IP la 1 mela accounts irukka nu check pannrom.

**Eg:** 192.168.0.1 IP la Gowtham, Ravi accounts create pannirukkanga ✅, Priya ip la single account mattum irukku ❌.

## Use Case Value (Tanglish)

Gowtham can use this to identify **potential duplicate or fraud accounts created from same IP for security checks.**

# 🔄 Problem 72: Identify Users Who Placed and Cancelled Orders Repeatedly

## 🧩 Question (Tanglish)

Identify users who **placed and cancelled orders repeatedly.**

## 🗂 Problem Statement (Tanglish)

Repeated-a order panni cancel pannina users identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE user\_orders ( order\_id INT,

user\_id INT,

user\_name VARCHAR(50), order\_status VARCHAR(20), order\_date DATE

);

INSERT INTO user\_orders VALUES

(1, 101, 'Gowtham', 'Cancelled', '2025-01-05'),

(2, 101, 'Gowtham', 'Cancelled', '2025-01-10'),

(3, 101, 'Gowtham', 'Cancelled', '2025-01-15'),

(4, 102, 'Arun', 'Completed', '2025-01-07');

## 💡 Solution (MySQL)

SELECT user\_id, user\_name, COUNT(\*) AS cancel\_count FROM user\_orders

WHERE order\_status = 'Cancelled' GROUP BY user\_id, user\_name HAVING cancel\_count >= 3;

## 🩻 Explanation (Tanglish)

* **WHERE order\_status = 'Cancelled':** Cancelled orders filter pannrom.
* **GROUP BY user\_id, user\_name:** User wise group pannrom.
* **HAVING cancel\_count >= 3:** 3 times mela cancel pannina users identify pannrom.

**Eg:** Gowtham 3 times cancel pannirukkanga ✅, Arun cancel pannala ❌.

## 🔄 Use Case Value (Tanglish)

Gowtham can use this to identify **repeat cancellation users for order policy refinement and fraud checks.**

# 🚖 Problem 73: Find Drivers Who Cancelled Rides at the Last Minute

## 🧩 Question (Tanglish)

Find drivers who **cancelled rides at the last minute.**

## 🗂 Problem Statement (Tanglish)

Last minute ride cancel pannina drivers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE driver\_rides ( ride\_id INT,

driver\_id INT,

driver\_name VARCHAR(50), cancel\_time\_minutes\_before INT, ride\_date DATE

);

INSERT INTO driver\_rides VALUES (1, 201, 'Sathish', 2, '2025-02-01'),

(2, 201, 'Sathish', 3, '2025-02-05'),

(3, 202, 'Kumar', 15, '2025-02-03');

## 💡 Solution (MySQL)

SELECT driver\_id, driver\_name, COUNT(\*) AS last\_minute\_cancels FROM driver\_rides

WHERE cancel\_time\_minutes\_before <= 5 GROUP BY driver\_id, driver\_name;

## 🩻 Explanation (Tanglish)

* **cancel\_time\_minutes\_before <= 5:** Last 5 mins kuLLa cancel pannina rides filter pannrom.
* **GROUP BY driver\_id, driver\_name:** Driver wise group pannrom.

**Eg:** Sathish last minute la 2 rides cancel pannirukkanga ✅, Kumar 15 mins munnaadi cancel pannirukkanga ❌.

## 🚖 Use Case Value (Tanglish)

Gowtham can use this to identify **last minute cancelling drivers for performance review.**

# 💸 Problem 74: Identify Customers Who Raised Refund Requests Frequently

## 🧩 Question (Tanglish)

Identify customers who **raised refund requests frequently.**

## 🗂 Problem Statement (Tanglish)

Refund requests frequent-a raise pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE refund\_requests ( request\_id INT,

customer\_id INT, customer\_name VARCHAR(50), request\_date DATE

);

INSERT INTO refund\_requests VALUES (1, 301, 'Gowtham', '2025-03-01'),

(2, 301, 'Gowtham', '2025-03-10'),

(3, 301, 'Gowtham', '2025-03-20'),

(4, 302, 'Meena', '2025-03-05');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS refund\_count FROM refund\_requests

GROUP BY customer\_id, customer\_name HAVING refund\_count >= 3;

## 🩻 Explanation (Tanglish)

* **COUNT >= 3:** 3 times mela refund request pannina customers identify pannrom.

**Eg:** Gowtham 3 refund requests pannirukkanga ✅, Meena 1 request pannirukkanga ❌.

## 💸 Use Case Value (Tanglish)

Gowtham can use this to identify **frequent refund seekers for policy monitoring and product quality checks.**

# 🛡 Problem 75: Identify Users with Suspicious Login Attempts

## 🧩 Question (Tanglish)

Gowtham wants to identify users with **suspicious login attempts.**

## 🗂 Problem Statement (Tanglish)

Suspicious login attempts panna users identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE user\_logins ( login\_id INT,

user\_id INT,

user\_name VARCHAR(50), login\_time DATETIME, is\_suspicious BOOLEAN

);

INSERT INTO user\_logins VALUES

(1, 401, 'Gowtham', '2025-04-01 02:00:00', TRUE),

(2, 401, 'Gowtham', '2025-04-02 03:00:00', TRUE),

(3, 402, 'Priya', '2025-04-01 10:00:00', FALSE);

## 💡 Solution (MySQL)

SELECT user\_id, user\_name, COUNT(\*) AS suspicious\_attempts FROM user\_logins

WHERE is\_suspicious = TRUE GROUP BY user\_id, user\_name HAVING suspicious\_attempts >= 2;

## 🩻 Explanation (Tanglish)

* **is\_suspicious = TRUE:** Suspicious login attempts filter pannrom.
* **COUNT >= 2:** 2 times mela suspicious login attempt pannina users filter pannrom.

**Eg:** Gowtham 2 suspicious logins pannirukkanga ✅, Priya illa ❌.

## 🛡 Use Case Value (Tanglish)

Gowtham can use this to identify **potential security risks for user accounts.**

# 🏠 Problem 76: Identify Customers Who Changed Addresses Frequently

## 🧩 Question (Tanglish)

Find customers who **changed addresses frequently.**

## 🗂 Problem Statement (Tanglish)

Address frequent-a maathina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE address\_changes ( change\_id INT,

customer\_id INT, customer\_name VARCHAR(50), address VARCHAR(100), change\_date DATE

);

INSERT INTO address\_changes VALUES (1, 501, 'Gowtham', 'Chennai', '2025-01-01'),

(2, 501, 'Gowtham', 'Coimbatore', '2025-03-01'),

(3, 501, 'Gowtham', 'Madurai', '2025-05-01'),

(4, 502, 'Keerthi', 'Chennai', '2025-02-01');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS address\_changes\_count FROM address\_changes

GROUP BY customer\_id, customer\_name HAVING address\_changes\_count >= 3;

## 🩻 Explanation (Tanglish)

* **COUNT >= 3:** 3 times mela address maathina customers filter pannrom.

**Eg:** Gowtham 3 times address maathirukkanga ✅, Keerthi 1 time mattum maathirukkanga ❌.

## 🏠 Use Case Value (Tanglish)

Gowtham can use this to identify **frequent movers for address verification and delivery risk analysis.**

# 💳 Problem 77: Identify Accounts with Same Payment Method Used Across Multiple IDs

## 🧩 Question (Tanglish)

Find accounts where **same payment method used across multiple user IDs.**

## 🗂 Problem Statement (Tanglish)

Oru payment method multiple user IDs la use pannina accounts identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE user\_payments ( payment\_id INT,

user\_id INT,

user\_name VARCHAR(50), payment\_method VARCHAR(50), payment\_date DATE

);

INSERT INTO user\_payments VALUES

(1, 601, 'Gowtham', 'Card XXXX1234', '2025-04-01'),

(2, 602, 'Ravi', 'Card XXXX1234', '2025-04-02'),

(3, 603, 'Anu', 'Card XXXX5678', '2025-04-03');

## 💡 Solution (MySQL)

SELECT payment\_method, COUNT(DISTINCT user\_id) AS user\_count FROM user\_payments

GROUP BY payment\_method HAVING user\_count > 1;

## 🩻 Explanation (Tanglish)

* **GROUP BY payment\_method:** Payment method wise group pannrom.
* **COUNT DISTINCT user\_id > 1:** Andha payment method multiple user IDs la use aagirtha nu filter pannrom.

**Eg:** Card XXXX1234 Gowtham and Ravi use pannirukkanga ✅, Card XXXX5678 Anu mattum use pannirukkanga ❌.

## 💳 Use Case Value (Tanglish)

Gowtham can use this to identify **shared payment methods for fraud detection and security checks.**

# 🛑 Problem 78: Identify Users with Multiple Failed Payment Attempts

## 🧩 Question (Tanglish)

Gowtham wants to identify users with **multiple failed payment attempts.**

## 🗂 Problem Statement (Tanglish)

Multiple failed payment attempts pannina users identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE payment\_attempts ( attempt\_id INT,

user\_id INT,

user\_name VARCHAR(50), status VARCHAR(20), attempt\_date DATE

);

INSERT INTO payment\_attempts VALUES (1, 701, 'Gowtham', 'Failed', '2025-05-01'),

(2, 701, 'Gowtham', 'Failed', '2025-05-02'),

(3, 701, 'Gowtham', 'Failed', '2025-05-03'),

(4, 702, 'Divya', 'Success', '2025-05-02');

## 💡 Solution (MySQL)

SELECT user\_id, user\_name, COUNT(\*) AS failed\_attempts FROM payment\_attempts

WHERE status = 'Failed' GROUP BY user\_id, user\_name HAVING failed\_attempts >= 3;

## 🩻 Explanation (Tanglish)

* **status = 'Failed':** Failed payment attempts filter pannrom.
* **COUNT >= 3:** 3 times mela failed attempts pannina users filter pannrom.

**Eg:** Gowtham 3 failed attempts pannirukkanga ✅, Divya success attempt pannirukkanga ❌.

## 🛑 Use Case Value (Tanglish)

Gowtham can use this to identify **payment issue users for alternate payment options and risk management.**

# 📦 Problem 79: Find Delivery Partners Who Marked Delivered but Customer Reported Missing

## 🧩 Question (Tanglish)

Find delivery partners who **marked orders as delivered but customer reported missing.**

## 🗂 Problem Statement (Tanglish)

Delivered nu mark pannitu customer "item missing" nu complain pannina delivery partners identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE delivery\_reports ( report\_id INT, delivery\_partner\_id INT,

partner\_name VARCHAR(50), status\_marked VARCHAR(20), customer\_complaint VARCHAR(20), delivery\_date DATE

);

INSERT INTO delivery\_reports VALUES

(1, 801, 'Suresh', 'Delivered', 'Missing', '2025-06-01'),

(2, 801, 'Suresh', 'Delivered', 'Missing', '2025-06-05'),

(3, 802, 'Raj', 'Delivered', 'Received', '2025-06-02');

## 💡 Solution (MySQL)

SELECT delivery\_partner\_id, partner\_name, COUNT(\*) AS missing\_reports FROM delivery\_reports

WHERE status\_marked = 'Delivered' AND customer\_complaint = 'Missing' GROUP BY delivery\_partner\_id, partner\_name

HAVING missing\_reports >= 2;

## 🩻 Explanation (Tanglish)

* **WHERE Delivered and Missing:** Delivered nu mark pannitu customer "missing" nu complain pannina cases filter pannrom.
* **COUNT >= 2:** 2 times mela nadandha partners filter pannrom.

**Eg:** Suresh 2 times nadandhurukkanga ✅, Raj la illa ❌.

## 📦 Use Case Value (Tanglish)

Gowtham can use this to identify **high-risk delivery partners for investigation and training.**

# 🎭 Problem 80: Identify Users Who Used Fake Promo Codes

## 🧩 Question (Tanglish)

Identify users who **used fake promo codes.**

## 🗂 Problem Statement (Tanglish)

Fake promo codes use pannina users identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE promo\_code\_usage ( usage\_id INT,

user\_id INT,

user\_name VARCHAR(50), promo\_code VARCHAR(50), is\_fake BOOLEAN, usage\_date DATE

);

INSERT INTO promo\_code\_usage VALUES

(1, 901, 'Gowtham', 'FREE100', TRUE, '2025-07-01'),

(2, 901, 'Gowtham', 'SAVE50', TRUE, '2025-07-05'),

(3, 902, 'Kavi', 'WELCOME10', FALSE, '2025-07-02');

## 💡 Solution (MySQL)

SELECT user\_id, user\_name, COUNT(\*) AS fake\_promo\_usage FROM promo\_code\_usage

WHERE is\_fake = TRUE GROUP BY user\_id, user\_name;

## 🩻 Explanation (Tanglish)

* **is\_fake = TRUE:** Fake promo code usage filter pannrom.
* **COUNT:** Fake promo code use pannina times kaanom.

**Eg:** Gowtham 2 fake promo codes use pannirukkanga ✅, Kavi illa ❌.

## 🎭 Use Case Value (Tanglish)

Gowtham can use this to identify **fraud promo code users for blocking or restriction.**

# 📈 Problem 81: Identify Customers with Consistent Purchase Patterns Month-on-Month

## 🧩 Question (Tanglish)

Identify customers with **consistent purchase patterns month-on-month.**

## 🗂 Problem Statement (Tanglish)

Month-on-month regular-a purchase pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE monthly\_purchases ( purchase\_id INT,

customer\_id INT, customer\_name VARCHAR(50), purchase\_date DATE

);

INSERT INTO monthly\_purchases VALUES (1, 1001, 'Gowtham', '2025-01-15'),

(2, 1001, 'Gowtham', '2025-02-15'),

(3, 1001, 'Gowtham', '2025-03-15'),

(4, 1002, 'Priya', '2025-01-20'),

(5, 1002, 'Priya', '2025-03-25');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(DISTINCT DATE\_FORMAT(purchase\_date, '%Y-%m')) AS active\_months

FROM monthly\_purchases

GROUP BY customer\_id, customer\_name HAVING active\_months >= 3;

## 🩻 Explanation (Tanglish)

* **DATE\_FORMAT(purchase\_date, '%Y-%m')** month-wise unique purchases check pannrom.
* **COUNT >= 3:** 3 months continuous-a purchase pannina customers filter pannrom.

**Eg:** Gowtham 3 months consistent purchase pannirukkanga ✅, Priya 2 months la pannirukkanga ❌.

## 📈 Use Case Value (Tanglish)

Gowtham can use this to identify **loyal customers for rewards and retention plans.**

# 🍂 Problem 82: Identify Customers with Seasonal Purchasing Behavior

## 🧩 Question (Tanglish)

Gowtham wants to find customers with **seasonal purchasing behavior.**

## 🗂 Problem Statement (Tanglish)

Specific seasons la than purchase panna virumbi pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE seasonal\_purchases ( purchase\_id INT,

customer\_id INT, customer\_name VARCHAR(50), purchase\_date DATE

);

INSERT INTO seasonal\_purchases VALUES (1, 1101, 'Karthik', '2025-12-15'),

(2, 1101, 'Karthik', '2024-12-20'),

(3, 1102, 'Anu', '2025-06-10');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS seasonal\_purchases FROM seasonal\_purchases

WHERE MONTH(purchase\_date) IN (11, 12)

GROUP BY customer\_id, customer\_name HAVING seasonal\_purchases >= 2;

## 🩻 Explanation (Tanglish)

* **MONTH(purchase\_date) IN (11,12):** Winter season la purchases filter pannrom.
* **COUNT >= 2:** 2 seasons la purchase pannina customers filter pannrom.

**Eg:** Karthik 2 winter seasons la purchase pannirukkanga ✅, Anu summer la mattum pannirukkanga ❌.

## 🍂 Use Case Value (Tanglish)

Gowtham can use this to identify **seasonal buyers for targeted seasonal campaigns.**

# ⏰ Problem 83: Find Customers Who Made Purchases on Specific Time Windows

## 🧩 Question (Tanglish)

Find customers who made purchases **within specific time windows (e.g., 6 PM - 9 PM).**

## 🗂 Problem Statement (Tanglish)

Specific time window la purchase pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE timed\_purchases ( purchase\_id INT,

customer\_id INT, customer\_name VARCHAR(50), purchase\_time TIME, purchase\_date DATE

);

INSERT INTO timed\_purchases VALUES

(1, 1201, 'Gowtham', '18:30:00', '2025-08-01'),

(2, 1201, 'Gowtham', '19:15:00', '2025-08-02'),

(3, 1202, 'Anjali', '15:00:00', '2025-08-01');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS evening\_purchases FROM timed\_purchases

WHERE purchase\_time BETWEEN '18:00:00' AND '21:00:00'

GROUP BY customer\_id, customer\_name;

## 🩻 Explanation (Tanglish)

* **purchase\_time BETWEEN '18:00:00' AND '21:00:00':** Evening time window la purchases filter pannrom.
* **COUNT:** Evening purchases count kaanom.

**Eg:** Gowtham evening time la 2 purchases pannirukkanga ✅, Anjali afternoon la mattum pannirukkanga ❌.

## ⏰ Use Case Value (Tanglish)

Gowtham can use this to identify **prime time buyers for targeted push notifications.**

# ☎ Problem 84: Identify Customers Who Interacted with Support Frequently

## 🧩 Question (Tanglish)

Identify customers who **interacted with customer support frequently.**

## 🗂 Problem Statement (Tanglish)

Customer support frequent-a contact pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE support\_interactions ( interaction\_id INT,

customer\_id INT, customer\_name VARCHAR(50), interaction\_date DATE

);

INSERT INTO support\_interactions VALUES (1, 1301, 'Gowtham', '2025-07-01'),

(2, 1301, 'Gowtham', '2025-07-05'),

(3, 1301, 'Gowtham', '2025-07-10'),

(4, 1302, 'Meena', '2025-07-02');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS support\_calls FROM support\_interactions

GROUP BY customer\_id, customer\_name HAVING support\_calls >= 3;

## 🩻 Explanation (Tanglish)

* **COUNT >= 3:** 3 times mela support contact pannina customers filter pannrom.

**Eg:** Gowtham 3 support interactions pannirukkanga ✅, Meena 1 interaction pannirukkanga ❌.

## ☎ Use Case Value (Tanglish)

Gowtham can use this to identify **high-touch customers for premium support plans or issue resolution.**

# 💹 Problem 85: Identify Customers with Increasing Order Value

## 🧩 Question (Tanglish)

Gowtham wants to find customers with **increasing order value over time.**

## 🗂 Problem Statement (Tanglish)

Order value time ku time increase aagura customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE order\_values ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), order\_value DECIMAL(10,2), order\_date DATE

);

INSERT INTO order\_values VALUES

(1, 1401, 'Gowtham', 500.00, '2025-06-01'),

(2, 1401, 'Gowtham', 750.00, '2025-07-01'),

(3, 1401, 'Gowtham', 1000.00, '2025-08-01'),

(4, 1402, 'Ravi', 800.00, '2025-06-05'),

(5, 1402, 'Ravi', 700.00, '2025-07-05');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name FROM order\_values

GROUP BY customer\_id, customer\_name HAVING MIN(order\_value) < MAX(order\_value);

## 🩻 Explanation (Tanglish)

* **MIN(order\_value) < MAX(order\_value):** Order value increase aagirucha nu check pannrom.

**Eg:** Gowtham oda order value increase aagiduchu ✅, Ravi oda decrease aagiruchu ❌.

## 💹 Use Case Value (Tanglish)

Gowtham can use this to identify **high-potential customers for premium upselling campaigns.**

# 📉 Problem 86: Identify Customers with Declining Activity Over Time

## 🧩 Question (Tanglish)

Find customers with **declining activity over time.**

## 🗂 Problem Statement (Tanglish)

Time pogum poguthu activity kuraindha customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE customer\_activity ( activity\_id INT,

customer\_id INT, customer\_name VARCHAR(50), activity\_count INT, activity\_month DATE

);

INSERT INTO customer\_activity VALUES (1, 1501, 'Gowtham', 10, '2025-05-01'),

(2, 1501, 'Gowtham', 7, '2025-06-01'),

(3, 1501, 'Gowtham', 4, '2025-07-01'),

(4, 1502, 'Kumar', 5, '2025-05-01'),

(5, 1502, 'Kumar', 7, '2025-06-01');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name FROM customer\_activity

GROUP BY customer\_id, customer\_name

HAVING MIN(activity\_count) < MAX(activity\_count);

## 🩻 Explanation (Tanglish)

* **MIN(activity\_count) < MAX(activity\_count):** Declining activity check panna or reverse logic pottu improve panna mudiyum depending on the test.

**Eg:** Gowtham oda activity decrease aagiruchu ✅, Kumar oda increase aagiruchu ❌.

## 📉 Use Case Value (Tanglish)

Gowtham can use this to identify **churn risk customers for re-engagement campaigns.**

# 🔄 Problem 87: Identify Users Dormant for 6 Months but Returned

## 🧩 Question (Tanglish)

Identify users who were **dormant for 6 months but returned later.**

## 🗂 Problem Statement (Tanglish)

1. months inactive irundhu apram return aana users identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE user\_logins\_return ( login\_id INT,

user\_id INT,

user\_name VARCHAR(50), login\_date DATE

);

INSERT INTO user\_logins\_return VALUES (1, 1601, 'Gowtham', '2024-12-01'),

(2, 1601, 'Gowtham', '2025-07-01'),

(3, 1602, 'Divya', '2025-05-01'),

(4, 1602, 'Divya', '2025-06-01');

## 💡 Solution (MySQL)

SELECT user\_id, user\_name FROM user\_logins\_return GROUP BY user\_id, user\_name

HAVING DATEDIFF(MAX(login\_date), MIN(login\_date)) >= 180;

## 🩻 Explanation (Tanglish)

* **DATEDIFF >= 180:** 6 months gap ku apram return aana users identify pannrom.

**Eg:** Gowtham 6 months gap apram return pannirukkanga ✅, Divya regular logins pannirukkanga ❌.

## 🔄 Use Case Value (Tanglish)

Gowtham can use this to identify **re-engaged users for welcome back offers.**

# 📢 Problem 88: Track Customers Who Share Referral Codes Frequently

## 🧩 Question (Tanglish)

Gowtham wants to track customers who **share referral codes frequently.**

## 🗂 Problem Statement (Tanglish)

Referral codes frequent-a share pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE referral\_shares ( share\_id INT,

customer\_id INT, customer\_name VARCHAR(50), referral\_code VARCHAR(20), share\_date DATE

);

INSERT INTO referral\_shares VALUES

(1, 1701, 'Gowtham', 'REF123', '2025-04-01'),

(2, 1701, 'Gowtham', 'REF123', '2025-04-05'),

(3, 1701, 'Gowtham', 'REF123', '2025-04-10'),

(4, 1702, 'Anu', 'REF456', '2025-04-03');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS share\_count FROM referral\_shares

GROUP BY customer\_id, customer\_name HAVING share\_count >= 3;

## 🩻 Explanation (Tanglish)

* **COUNT >= 3:** 3 times mela referral codes share pannina customers filter pannrom.

**Eg:** Gowtham 3 times referral code share pannirukkanga ✅, Anu 1 time mattum share pannirukkanga ❌.

## 📢 Use Case Value (Tanglish)

Gowtham can use this to identify **referral champions for rewards and tier promotions.**

# 🛠 Problem 89: Identify Customers Who Preferred Specific Payment Methods

## 🧩 Question (Tanglish)

Identify customers who consistently preferred a specific payment method (e.g., Credit Card).

## 🗂 Problem Statement (Tanglish)

Oru specific payment method mattum than use pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE payment\_methods ( payment\_id INT,

customer\_id INT, customer\_name VARCHAR(50), payment\_method VARCHAR(50), payment\_date DATE

);

INSERT INTO payment\_methods VALUES

(1, 8001, 'Gowtham', 'Credit Card', '2025-01-15'),

(2, 8001, 'Gowtham', 'Credit Card', '2025-02-20'),

(3, 8002, 'Anu', 'Debit Card', '2025-03-10'),

(4, 8002, 'Anu', 'Credit Card', '2025-03-15');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name FROM payment\_methods

GROUP BY customer\_id, customer\_name

HAVING SUM(payment\_method != 'Credit Card') = 0;

## 🩻 Explanation (Tanglish)

* **SUM(payment\_method != 'Credit Card') = 0:** Credit card illa payments irukka customers filter pannrom.
* **GROUP BY customer\_id, customer\_name:** Customer wise group pannrom.

**Example:** Gowtham ellam Credit Card use pannirukkanga ✅, Anu oru Debit Card use pannirukkanga ❌.

## 🛠 Use Case Value (Tanglish)

Gowtham can use this to identify **loyal payment method users for payment-specific offers.**

# 📦 Problem 90: Identify Customers Who Purchased High-Value Items on EMI

## 🧩 Question (Tanglish)

Find customers who purchased **high-value items on EMI.**

## 🗂 Problem Statement (Tanglish)

EMI-la high-value items (e.g., ₹50,000 mele) purchase pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE emi\_purchases ( purchase\_id INT,

customer\_id INT, customer\_name VARCHAR(50), order\_value DECIMAL(10, 2), payment\_mode VARCHAR(20), purchase\_date DATE

);

INSERT INTO emi\_purchases VALUES

(1, 9001, 'Gowtham', 75000.00, 'EMI', '2025-06-01'),

(2, 9002, 'Kavi', 35000.00, 'EMI', '2025-06-10'),

(3, 9003, 'Ravi', 60000.00, 'Credit Card', '2025-06-15');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name FROM emi\_purchases

WHERE payment\_mode = 'EMI' AND order\_value > 50000 GROUP BY customer\_id, customer\_name;

## 🩻 Explanation (Tanglish)

* **payment\_mode = 'EMI' AND order\_value > 50000:** EMI payment and high-value order filter pannrom.
* **GROUP BY customer\_id, customer\_name:** Customer wise group pannrom.

**Example:** Gowtham ₹75,000 EMI use pannirukkanga ✅, Kavi ₹35,000 EMI use pannirukkanga

❌.

## 📦 Use Case Value (Tanglish)

Gowtham can use this to identify **premium EMI customers for special EMI schemes and upsell.**

# 🏙 Problem 91: Gowtham Wants to Find Top 100 Spenders in Each City

## 🧩 Question (Tanglish)

Gowtham wants to find **top 100 spenders in each city.**

## 🗂 Problem Statement (Tanglish)

Oru city-la top 100 spending pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE city\_spenders ( customer\_id INT, customer\_name VARCHAR(50), city VARCHAR(50),

total\_spent DECIMAL(10,2)

);

INSERT INTO city\_spenders VALUES (1, 'Gowtham', 'Chennai', 50000.00),

(2, 'Arun', 'Chennai', 45000.00),

(3, 'Kavitha', 'Coimbatore', 60000.00),

(4, 'Suresh', 'Chennai', 48000.00),

(5, 'Anu', 'Coimbatore', 40000.00);

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, city, total\_spent FROM (

SELECT customer\_id, customer\_name, city, total\_spent,

ROW\_NUMBER() OVER (PARTITION BY city ORDER BY total\_spent DESC) AS rn FROM city\_spenders

) t

WHERE rn <= 100;

## 🩻 Explanation (Tanglish)

### ROW\_NUMBER() OVER (PARTITION BY city ORDER BY total\_spent DESC):

City-wise ranking create pannrom.

* **WHERE rn <= 100:** Top 100 spenders filter pannrom.

**Eg:** Chennai-la Gowtham top spender-a irukkanga ✅, city-wise top 100 kudutha output varum.

## 🌎 Use Case Value (Tanglish)

Gowtham and company can use this to target **high-spending customers city-wise for premium offers and loyalty rewards.**

# 🤝 Problem 92: Identify Customers Who Referred Friends Frequently

## 🧩 Question (Tanglish)

Identify customers who **referred friends frequently.**

## 🗂 Problem Statement (Tanglish)

Frequent-a friends refer pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE referral\_records ( referral\_id INT,

customer\_id INT, customer\_name VARCHAR(50), referred\_friend\_id INT,

referral\_date DATE

);

INSERT INTO referral\_records VALUES (1, 1001, 'Gowtham', 2001, '2025-01-10'),

(2, 1001, 'Gowtham', 2002, '2025-02-15'),

(3, 1001, 'Gowtham', 2003, '2025-03-20'),

(4, 1002, 'Anu', 2004, '2025-02-10');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS referral\_count FROM referral\_records

GROUP BY customer\_id, customer\_name HAVING referral\_count >= 3;

## 🩻 Explanation (Tanglish)

* **COUNT >= 3:** 3 times mela friends refer pannina customers filter pannrom.

**Eg:** Gowtham 3 friends refer pannirukkanga ✅, Anu 1 friend mattum refer pannirukkanga ❌.

## 🤝 Use Case Value (Tanglish)

Gowtham can use this to identify **top referrers for rewards and referral campaigns.**

# 📉 Problem 93: Find Users Who Churned After High Activity

## 🧩 Question (Tanglish)

Find users who **churned (inactive) after high activity.**

## 🗂 Problem Statement (Tanglish)

High activity panna apram churn aana users identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE user\_activity ( activity\_id INT,

user\_id INT,

user\_name VARCHAR(50), activity\_level VARCHAR(20), last\_active\_date DATE

);

INSERT INTO user\_activity VALUES

(1, 3001, 'Gowtham', 'High', '2025-01-01'),

(2, 3002, 'Ravi', 'Low', '2025-06-01'),

(3, 3003, 'Sahana', 'High', '2024-09-01');

💡 **Solution (MySQL)** SELECT user\_id, user\_name FROM user\_activity

WHERE activity\_level = 'High' AND last\_active\_date < DATE\_SUB(CURDATE(), INTERVAL 3 MONTH);

## 🩻 Explanation (Tanglish)

* **activity\_level = 'High':** High activity users filter pannrom.
* **last\_active\_date < 3 months ago:** 3 months inactive users filter pannrom.

**Eg:** Gowtham last active 6 months munnaadi, churn aagirukkanga ✅, Ravi low activity users inactive irukkanga ❌.

## 📉 Use Case Value (Tanglish)

Gowtham can use this to identify **high potential churn users for re-engagement campaigns.**

Adding Problems 94 and 95 inside your book document now, Gowtham:

# 🛍 Problem 94: Identify Customers with High Basket Size but Low Purchase Frequency

## 🧩 Question (Tanglish)

Identify customers who have **high basket size but low purchase frequency.**

## 🗂 Problem Statement (Tanglish)

Periya order size irundhalum, purchase frequency kuraiya irundha customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE customer\_orders ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), basket\_size INT,

order\_date DATE

);

INSERT INTO customer\_orders VALUES (1, 4001, 'Gowtham', 15, '2025-01-01'),

(2, 4001, 'Gowtham', 20, '2025-06-01'),

(3, 4002, 'Anu', 5, '2025-02-01'),

(4, 4002, 'Anu', 6, '2025-02-15');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name FROM (

SELECT customer\_id, customer\_name, AVG(basket\_size) AS avg\_basket, COUNT(\*) AS order\_count

FROM customer\_orders

GROUP BY customer\_id, customer\_name

) t

WHERE avg\_basket >= 10 AND order\_count <= 2;

## 🩻 Explanation (Tanglish)

* **AVG(basket\_size) >= 10:** Periya order size customers filter pannrom.
* **order\_count <= 2:** Kuraiya purchase frequency filter pannrom.

**Eg:** Gowtham periya basket size with low frequency ✅, Anu frequent but small basket size ❌.

## 🛍 Use Case Value (Tanglish)

Gowtham can use this to identify **big spenders with infrequent purchases for targeted campaigns.**

# 🔍 Problem 95: Identify Customers with the Most Diverse Purchases

## 🧩 Question (Tanglish)

Find customers with the **most diverse product categories purchased.**

## 🗂 Problem Statement (Tanglish)

Various product categories purchase pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE customer\_purchases ( purchase\_id INT,

customer\_id INT, customer\_name VARCHAR(50), product\_category VARCHAR(50)

);

INSERT INTO customer\_purchases VALUES (1, 5001, 'Gowtham', 'Electronics'),

(2, 5001, 'Gowtham', 'Books'),

(3, 5001, 'Gowtham', 'Clothing'),

(4, 5002, 'Ravi', 'Electronics'),

(5, 5002, 'Ravi', 'Electronics');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(DISTINCT product\_category) AS category\_count

FROM customer\_purchases

GROUP BY customer\_id, customer\_name ORDER BY category\_count DESC

LIMIT 10;

## 🩻 Explanation (Tanglish)

* **COUNT(DISTINCT product\_category):** Product category diversity count pannrom.
* **ORDER BY category\_count DESC LIMIT 10:** Top 10 diverse purchasers filter pannrom.

**Eg:** Gowtham 3 categories purchase pannirukkanga ✅, Ravi single category mattum pannirukkanga ❌.

## 🔍 Use Case Value (Tanglish)

Gowtham can use this to identify **diverse shoppers for cross-selling and personalized offers.**

# 📊 Problem 96: Find Customers with the Highest Return Rates

## 🧩 Question (Tanglish)

Find customers who have the **highest product return rates.**

## 🗂 Problem Statement (Tanglish)

Maximum product returns pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE orders ( order\_id INT, customer\_id INT,

customer\_name VARCHAR(50), is\_returned BOOLEAN

);

INSERT INTO orders VALUES (1, 6001, 'Gowtham', TRUE),

(2, 6001, 'Gowtham', TRUE),

(3, 6002, 'Anu', FALSE),

(4, 6002, 'Anu', TRUE);

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, SUM(is\_returned) / COUNT(\*) AS return\_rate

FROM orders

GROUP BY customer\_id, customer\_name ORDER BY return\_rate DESC

LIMIT 10;

## 🩻 Explanation (Tanglish)

* **SUM(is\_returned) / COUNT(\*)** return rate calculate pannrom.
* **ORDER BY return\_rate DESC** highest return rate customers top la varum.

**Eg:** Gowtham return rate adhigam ✅, Anu adhigama illa ❌.

## 📊 Use Case Value (Tanglish)

Gowtham can use this to identify **potential problem customers for quality control and targeted support.**

# 📞 Problem 97: Identify Customers Who Availed Maximum Customer Support

## 🧩 Question (Tanglish)

Identify customers who **availed maximum customer support interactions.**

## 🗂 Problem Statement (Tanglish)

Maximum support calls pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE support\_calls ( call\_id INT,

customer\_id INT, customer\_name VARCHAR(50)

);

INSERT INTO support\_calls VALUES (1, 7001, 'Gowtham'),

(2, 7001, 'Gowtham'),

(3, 7001, 'Gowtham'),

(4, 7002, 'Kavi');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS call\_count FROM support\_calls

GROUP BY customer\_id, customer\_name ORDER BY call\_count DESC

LIMIT 10;

## 🩻 Explanation (Tanglish)

* **COUNT(\*) AS call\_count** customer support calls count pannrom.
* **ORDER BY call\_count DESC** max calls panna customers top la varum.

**Eg:** Gowtham 3 support calls pannirukkanga ✅, Kavi 1 mattum pannirukkanga ❌.

## 📞 Use Case Value (Tanglish)

Gowtham can use this to identify **customers needing extra support for personalized service.**

Adding Problems 98 and 99 inside your book now, Gowtham:

# 🌙 Problem 98: Gowtham Wants to Find Customers Who Did Late-Night Shopping Frequently

## 🧩 Question (Tanglish)

Gowtham wants to find customers who did **late-night shopping frequently.**

## 🗂 Problem Statement (Tanglish)

Raatri 10 PM appuram frequent-a shopping pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE late\_night\_shopping ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), order\_time TIME

);

INSERT INTO late\_night\_shopping VALUES (1, 8001, 'Gowtham', '22:30:00'),

(2, 8001, 'Gowtham', '23:00:00'),

(3, 8001, 'Gowtham', '22:45:00'),

(4, 8002, 'Anu', '21:30:00');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name, COUNT(\*) AS late\_night\_orders FROM late\_night\_shopping

WHERE order\_time >= '22:00:00'

GROUP BY customer\_id, customer\_name HAVING late\_night\_orders >= 3;

## 🩻 Explanation (Tanglish)

* **order\_time >= '22:00:00'** late night orders filter pannrom.
* **COUNT >= 3** frequent late night shopping customers identify pannrom.

**Eg:** Gowtham frequent-a late night shopping pannirukkanga ✅, Anu mattum evening time mattum pannirukkanga ❌.

## 🌙 Use Case Value (Tanglish)

Gowtham can use this to target **late-night shoppers for special offers and notifications.**

# ❌ Problem 99: Identify Customers Who Placed the First Order and Never Returned

## 🧩 Question (Tanglish)

Identify customers who **placed the first order and never returned.**

## 🗂 Problem Statement (Tanglish)

First order pannitu marubadiyum purchase pannatha customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE customer\_orders ( order\_id INT,

customer\_id INT, customer\_name VARCHAR(50), order\_date DATE

);

INSERT INTO customer\_orders VALUES (1, 9001, 'Gowtham', '2025-01-10'),

(2, 9002, 'Anu', '2025-02-15'),

(3, 9002, 'Anu', '2025-03-20');

## 💡 Solution (MySQL)

SELECT customer\_id, customer\_name

FROM customer\_orders

GROUP BY customer\_id, customer\_name HAVING COUNT(\*) = 1;

## 🩻 Explanation (Tanglish)

* **COUNT(\*) = 1** oru order mattum pannirukkira customers identify pannrom.

**Eg:** Gowtham first order mattum pannirukkanga ✅, Anu rendu order pannirukkanga ❌.

## ❌ Use Case Value (Tanglish)

Gowtham can use this to identify **one-time buyers for retention campaigns.**

Adding Problem 100 inside your book now, Gowtham:

# 🎯 Problem 100: Identify Customers Who Engaged in All Platform Features

## 🧩 Question (Tanglish)

Find customers who **engaged in all platform features** (e.g., orders, reviews, referrals, support).

## 🗂 Problem Statement (Tanglish)

Orders, reviews, referrals, support ellathayum use pannina customers identify panna vendiyathu.

## 🛠 Create & Insert DDL (MySQL)

CREATE TABLE customer\_orders ( customer\_id INT, customer\_name VARCHAR(50)

);

CREATE TABLE customer\_reviews ( customer\_id INT,

customer\_name VARCHAR(50)

);

CREATE TABLE customer\_referrals ( customer\_id INT,

customer\_name VARCHAR(50)

);

CREATE TABLE customer\_support ( customer\_id INT,

customer\_name VARCHAR(50)

);

INSERT INTO customer\_orders VALUES (1, 'Gowtham'), (2, 'Anu'), (3, 'Ravi'); INSERT INTO customer\_reviews VALUES (1, 'Gowtham'), (2, 'Anu');

INSERT INTO customer\_referrals VALUES (1, 'Gowtham'), (3, 'Ravi'); INSERT INTO customer\_support VALUES (1, 'Gowtham'), (2, 'Anu');

## 💡 Solution (MySQL)

SELECT o.customer\_id, o.customer\_name FROM customer\_orders o

JOIN customer\_reviews r ON o.customer\_id = r.customer\_id JOIN customer\_referrals f ON o.customer\_id = f.customer\_id JOIN customer\_support s ON o.customer\_id = s.customer\_id;

## 🩻 Explanation (Tanglish)

* **JOIN ellam features tables:** Orders, reviews, referrals, support ellam join pannrom.
* **JOIN filter pannum customers only:** Ella features um use pannina customers maathiri filter pannrom.

**Eg:** Gowtham ellam features use pannirukkanga ✅, Anu oru oru feature miss pannirukkanga

❌.

## 🎯 Use Case Value (Tanglish)

Gowtham can use this to identify **power users for VIP treatment and special campaigns.**

# 📚 Final Notes and Conclusion

Dear readers,

Indha book la naan unga kitta **100 hard level SQL problems** share pannirukken, ella problems-um **real-world use cases la ninnum eduththu**, especially **FAANG, MAANG maadhiri top companies use cases** pola design pannirukku.

Unga SQL skills-ai next level ku edukka, **scenario-based practical problems** romba

mukkiyam. Indha problems solve pannitu, neenga real-world interviews, projects, and daily tasks la nalla ready-a iruppinga.

## Key Points:

* Ellam **MySQL flavor** la irukku, so simple-a easy-a follow panna mudiyum.
* Query explanation ellam **step-by-step, query by query**, easy Tanglish la irukku for better understanding.
* Company use cases la ninnu design pannirukken, so indha problems unga **career growth-ku direct-a help pannum**.
* Neenga oru fresher-um experience professional-um irundhaalum, indha book unga kuvara pathi cover pannirukku.
* Practice, patience, and continuous learning veendum. SQL is a vast language, but indha problems with proper explanations unga journey-a easy-a pannum.

## Ungaluku Enna Panradhu?

* Intha book la irukkara problems-ai **solution-um explanations-um purinjukonga**.
* **Practice pannunga** – concept clear aagum, and speed improve aagum.
* Interview prep-ku use pannunga.
* Real-world projects-ku ithai reference-a edunga.
* Social media la unga progress share pannunga – idhu unga personal branding-ku help pannum.

## Final Words

SQL oru **powerful tool**. Idhai master panna, neenga data world-la **valuable asset**

aagiruppinga. Indha book unga **SQL mastery-ku oru solid foundation**.

Naan unga success-ku wish pannuren! Edhaavadhu doubts irundhaal, illa next topics venumna sollunga. Naan udane help pannaren.

### Thank you so much for reading! Happy Querying!

— Gowtham

# 👤 About the Author: Gowtham SB – Data Engineer, Educator, Creator

Gowtham is one of India’s most loved **Tamil tech content creators** — known for explaining

**Data Engineering, System Design, and AI** using **local analogies** that stick.

* 💻 10+ years of experience in Big Data & Cloud
* 🎥 Creator of the YouTube channel: [**Data Engineering**](https://www.youtube.com/%40dataengineeringvideos)
* 📚 Author of multiple bestselling books on data & AI
* 🧠 Featured in IBM’s **Top Data Engineering Influencers** list
* 🌍 Over **2 lakh learners** across YouTube, LinkedIn, and workshops

He believes tech should be taught like a conversation at a tea kada — warm, real, and unforgettable.

You can connect with him on:

* YouTube: [Data Engineering](https://www.youtube.com/%40dataengineeringvideos)
* Instagram: @dataengineeringtamil
* LinkedIn: <https://www.linkedin.com/in/sbgowtham/>
* Website - <https://www.dataengineeringtamil.com/>
* My Books - https://topmate.io/dataengineering

**Thank You**