Part 1: Real App Use Cases

- **Amazon**:
  - Calculates the total number of orders per customer using COUNT(OrderID).
  - Identifies top-selling products by summing quantities and grouping by product.
- **Talabat / Uber Eats**:
  - Shows monthly restaurant earnings using SUM(OrderTotal) grouped by restaurant and month.
  - Calculates average rating per food item with AVG(Rating).
- **YouTube / Udemy**:
  - Tracks views per channel using SUM(Views) grouped by ChannelID.
  - Most-watched courses are found using COUNT(Views) per course.
  - Completion rates are calculated using AVG(CompletionPercent).
- **Dashboards**:
  - Show COUNT(*) for active users.
  - Use SUM(Sales) to track revenue growth.
  - Highlight top employees via COUNT(TasksCompleted) or performance metrics.

Part 2: Different Uses of Aggregation

1. **GROUP BY vs ORDER BY**:
   - GROUP BY groups rows for aggregation (e.g., AVG, SUM).
   - ORDER BY sorts the result set (ascending/descending).
2. **HAVING vs WHERE**:
   - WHERE filters rows before aggregation.
   - HAVING filters grouped/aggregated results.
3. **Common Mistakes**:

- Using columns in SELECT without grouping or aggregation.
- Misplacing HAVING instead of WHERE and vice versa.
- Forgetting to join relevant tables before aggregation.

4. **Using COUNT(DISTINCT...), AVG(...), SUM(...) Together**:
   - Use in dashboards or performance analytics. E.g., per student:
     - Total courses (COUNT(DISTINCT CourseID)),
     - Avg rating (AVG(Rating)),
     - Total spent (SUM(Price)).
5. **Performance & Indexes**:
   - GROUP BY can be slow on large tables.
   - Indexes on grouped columns help by allowing faster lookups and sorting.