**How do you understand the data when you first receive it?**  
or  
**What steps do you take before starting data processing?**

**🔑 Key Points to Cover in Your Answer:**

1. **Understand Business Context**
   * Why is the data collected?
   * What problem are we solving with it?
   * Example: *Sales data → used for revenue forecasting.*
2. **Explore Metadata / Data Dictionary**
   * Check **schema, column names, data types**.
   * Ask: Does a “date” field actually contain proper dates? Is “amount” stored as integer, float, or string?
3. **Check Data Quality**
   * Missing values (NULLs, blanks).
   * Duplicates.
   * Incorrect formats (e.g., date as text “12/31/2025” vs “2025-12-31”).
   * Outliers (e.g., salary = 1,000,000,000).
4. **Distribution & Summary Statistics**
   * Run SQL: SELECT COUNT(\*), MIN(), MAX(), AVG() …
   * Identify skewness or unusual ranges.
   * Example: *Customer age column has values 200 → invalid.*
5. **Relationships Between Tables**
   * Check primary keys & foreign keys.
   * Verify join conditions (e.g., does every order have a matching customer?).
6. **Data Volume & Growth**
   * How many rows today?
   * How fast is it growing daily?
   * Important for pipeline scalability & partitioning strategy.
7. **Business Validation**
   * Compare with real-world expectations.
   * Example: *If company revenue is 10M, but dataset shows only 1M, something is missing.*

**🎯 Example Answer (**Data Engineer Style**)**

*When I receive a new dataset, I first try to understand the* ***business context****—what problem this data is solving. Then I explore the* ***schema and metadata*** *to understand columns, data types, and relationships. I check* ***data quality*** *by looking for missing values, duplicates, and outliers. I also run* ***summary statistics*** *to see distributions and ranges. For multi-table datasets, I validate* ***keys and relationships:*** *- to ensure joins are correct. Finally, I cross-check with* ***business expectations: -*** *to confirm that the dataset represents reality. This way, I avoid building pipelines on top of bad or misunderstood data.*

**🔹 1. How do you approach a new dataset you’ve never seen before?**

**Answer:**  
*"I start with the business context, then explore the schema to check column names, data types, and constraints. I run basic profiling queries for missing values, duplicates, and outliers. After that, I analyze distributions and validate if the dataset aligns with real business expectations."*

**🔹 2. How do you check data quality?**

**Answer:**  
*"I look for missing or null values, duplicates, incorrect formats, and outliers. I also validate primary key uniqueness and referential integrity between tables. Automated profiling tools like Great Expectations or SQL checks help maintain quality in pipelines."*

**🔹 3. What would you do if you find NULL or missing values?**

**Answer:**  
*"It depends on the use case. Options include removing rows, imputing values (mean, median, forward fill), or flagging them. For critical fields like primary keys, I’d investigate upstream sources to fix the root cause."*

**🔹 4. How do you validate if data is correct or trustworthy?**

**Answer:**  
*"I compare dataset totals with external references, like financial reports or transaction counts. I also check whether values fall within realistic ranges and confirm with business SMEs if anomalies are valid or errors."*

**🔹 5. What’s the difference between data profiling and data exploration?**

**Answer:**  
*"Data profiling is a systematic check of metadata, distributions, and quality metrics. Data exploration is a deeper, business-driven analysis to find insights, trends, or patterns."*

**🔹 6. How do you check relationships between multiple tables?**

**Answer:**  
*"I validate primary and foreign keys, check for orphan records, and run join tests to ensure relationships hold. For example, every order should map to an existing customer in the customer table."*

**🔹 7. What are some key statistics you look at while understanding data?**

**Answer:**  
*"Count, distinct count, min, max, average, median, standard deviation, and percentiles. These help detect anomalies like negative sales or unrealistic ages."*

**🔹 8. How do you detect outliers in data?**

**Answer:**  
*"Through statistical checks (values beyond 3 standard deviations), visualizations (box plots, histograms), or business thresholds (e.g., salary > 10M is unusual)."*

**🔹 9. What if the data is too large to analyze directly?**

**Answer:**  
*"I’d sample the data to get an overview, use partitioned queries, or leverage distributed tools like Spark/BigQuery. Profiling can also be done on smaller subsets before scaling."*

**🔹 10. How do you ensure scalability when working with large datasets?**

**Answer:**  
*"I check the volume, velocity, and expected growth of data. Based on that, I design partitioning, indexing, or file format choices like Parquet/ORC for efficient querying."*

**🔹 11. Can you give an example of catching an error while understanding data?**

**Answer:**  
*"Yes, in a sales dataset, I noticed product prices were negative. On checking upstream, it turned out returns were being stored in the same column without a proper flag. I worked with the team to separate return transactions."*

**🔹 12. What’s the importance of understanding data before building pipelines?**

**Answer:**  
*"If we don’t understand the data, we risk building pipelines that produce incorrect or incomplete outputs. Early validation saves time, prevents rework, and ensures business trust in analytics."*

**🔹 Definition of Outliers**

An **outlier** is a data point that is **significantly different** from the majority of the dataset.  
It’s a value that falls **far outside the expected range or pattern** of the data.

👉 Example: If most employees’ ages are between **22–60**, but you find an age of **200**, that’s an outlier.

**🔹 Why Outliers Happen**

* **Data entry errors** → e.g., salary entered as 10000000 instead of 100000.
* **Measurement errors** → sensor glitches, wrong units.
* **Natural extreme values** → some customers genuinely spend 10x more than average.
* **Fraud or anomalies** → outliers can reveal fraud (e.g., abnormal credit card transactions).

**🔹 How to Detect Outliers**

1. **Simple Range Check (Business Rule):**
   * Example: Age should be 0–120. Anything outside is an outlier.
2. **Statistical Methods:**
   * **Z-score**:

Z=(x−μ)σZ = \frac{(x - \mu)}{\sigma}Z=σ(x−μ)​

If |Z| > 3 → outlier.

* + **IQR (Interquartile Range) Method:**
    - Q1 = 25th percentile, Q3 = 75th percentile
    - IQR = Q3 – Q1
    - Outlier if:

x<Q1−1.5×IQRorx>Q3+1.5×IQRx < Q1 - 1.5 \times IQR \quad \text{or} \quad x > Q3 + 1.5 \times IQRx<Q1−1.5×IQRorx>Q3+1.5×IQR

1. **Visualization:**
   * Boxplots, scatter plots, histograms highlight extreme values.

**🔹 What to Do with Outliers**

* **Investigate root cause** (data error vs valid case).
* **Remove** them if they are errors.
* **Cap/transform** values (winsorization, log transformation).
* **Keep them** if they are genuine and important (e.g., fraud detection).

**🎯 Sample Interview Answer**

*"An outlier is a data point that deviates significantly from other observations in a dataset. For example, if most transactions are between $100–$500, but one is $50,000, that’s an outlier. Outliers can occur due to* ***errors, anomalies, or genuine rare cases****. I usually detect them using statistical methods like Z-scores,* ***IQR****, or visualizations like boxplots. Depending on the context, I either investigate, remove, or keep them if they provide valuable insights, such as fraud detection."*

**Data Understanding Skills**

* Data types (categorical, numerical, time-series)
* Data quality issues (nulls, duplicates, outliers)
* Business metrics (sales, revenue, KPIs)

**Analytical Thinking**

* Ask the right questions: *Why did sales drop?*
* Use logic to connect data with business needs.

**Communication Skills**

* Write clear reports and explain data simply to non-technical people.
* Example: Instead of “variance -0.2 L,” say “We sold cars slightly below the expected price.”

**🚀 Simple Way to Remember**

A Data Analyst should always think in 4 steps:  
**Ask → Prepare → Analyze → Present**

* **Ask** → What is the business problem?
* **Prepare** → Collect and clean the right data.
* **Analyze** → Explore trends and patterns.
* **Present** → Show results with visuals and recommendations.

**📌 What is IQR?**

**IQR = Interquartile Range**  
It measures the **spread of the middle 50% of the data**.

👉 Formula:

IQR=Q3−Q1IQR = Q3 - Q1IQR=Q3−Q1

* **Q1 (1st Quartile / 25th percentile)** → Value at which 25% of the data lies below.
* **Q3 (3rd Quartile / 75th percentile)** → Value at which 75% of the data lies below.
* **IQR** → The difference between Q3 and Q1.

**📊 Example**

Dataset (sorted):  
2, 4, 5, 7, 8, 10, 12, 15, 18, 20

1. **Q1 (25th percentile)** = 5
2. **Q3 (75th percentile)** = 15
3. **IQR = Q3 – Q1 = 15 – 5 = 10**

So, the **middle 50% of data** lies between 5 and 15.

**✅ 1. Strengthen What You Already Know**

* Be ready with **examples** (don’t just know syntax, think how to use it in real data problems).  
  Example: *Find top 3 most expensive cars by brand using ROW\_NUMBER()*

**✅ 2. Study These Next (High-Value Topics for Interview)**

1. **Subqueries (Correlated & Non-Correlated)**
   * Example: *Find cars whose price is above the average price of all cars.*
2. **CTEs (Common Table Expressions)**
   * Good for stepwise problem solving → often asked in case studies.
3. **Case When (Conditional Logic)**
   * Example: *Classify cars as “Budget / Mid / Premium” based on Price\_Lakh.*
4. **Date & Time Functions** (important for analysts)
   * DATEDIFF, DATEADD, YEAR, MONTH, GETDATE()
   * Example: *Find cars with insurance expired in last 30 days.*
5. **Data Cleaning Functions**
   * TRIM, UPPER, LOWER, REPLACE, handling NULLs

**✅ 3. Focus on Problem-Solving Scenarios**

Interviewers don’t only check syntax—they check **thinking**.  
Be ready for questions like:

* Find the **top 3 account\_type** with the highest total\_balance.
* Count how many **cars are unsold**.
* Calculate **average service cost per car**.
* Find the **most recent sale date per city**.

**✅ 4. Basics Beyond SQL (Analyst Role)**

* **Power BI / Visualization thinking** → KPIs, dashboards (Sales, Unsold Cars, Market Value).
* **Statistics basics**: Mean, Median, Mode, IQR (you asked already 👍).
* **Business metrics**: Revenue, Variance, Growth %, Conversion Rate.

------------------------------------------------------------------------------------------------------------------------------------------------

**1️⃣ Advanced SQL / Interview-Frequent Topics (High Priority)**

Even if you know joins, aggregations, windows, etc., make sure you **can write them fast under pressure**:

**a) CASE Statements**

* Conditional logic inside SELECT.
* Example: CASE WHEN sales > 1000 THEN 'High' ELSE 'Low' END AS sales\_category
* Often asked for transforming categorical data.

**b) Subqueries**

* **Scalar subquery**, **correlated subquery**, and **in WHERE/HAVING**.
* Example: "Find employees whose salary is above the average salary."

**c) Combining Window Functions with Aggregations**

* Example: SUM(sales) OVER(PARTITION BY region ORDER BY month ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW)
* Shows your ability to do cumulative calculations.

**d) CTE + Recursive Queries (if advanced level)**

* You know CTE basics, but recursive CTE is sometimes asked:

WITH RECURSIVE numbers AS (

SELECT 1 AS n

UNION ALL

SELECT n + 1 FROM numbers WHERE n < 10

)

SELECT \* FROM numbers;

**2️⃣ Data Cleaning / Null Handling**

* You already know IFNULL, COALESCE, NULLIF.
* Also prepare:
  + Handling duplicates: DISTINCT vs ROW\_NUMBER() OVER(PARTITION BY …) for de-duplication.
  + Filling missing values (mean, median, mode) — can explain conceptually if asked.

**3️⃣ Short Analytical Questions**

* You may get **scenario-based questions**:
  1. Find top 3 customers by revenue per region. ✅ Window functions
  2. Find second highest salary. ✅ RANK / DENSE\_RANK
  3. Find months with below-average sales. ✅ Aggregations + HAVING
  4. Remove duplicate entries keeping latest date. ✅ ROW\_NUMBER()

Tip: Think **“how would I write the SQL in 2-3 steps?”**

**4️⃣ Quick Excel / BI Knowledge (Optional)**

* Basic pivot tables, VLOOKUP/XLOOKUP, conditional formatting.
* If asked about dashboards: **Power BI / Tableau** — explain filters, visuals, KPI metrics.

**5️⃣ Interview Strategy**

* **Strong points:** Joins, windows, aggregations, filters → you are already confident.
* **Revise tonight:** CASE, subqueries, handling duplicates, top-N / ranking queries.
* **Practice:** 5-6 queries end-to-end using small sample datasets.
* **Explain logic clearly**: Interviewers care more about your **approach** than typing speed.

✅ **Priority to cover before tomorrow:**

1. CASE statements + conditional logic
2. Subqueries (correlated & scalar)
3. De-duplication techniques (ROW\_NUMBER)
4. Top-N queries (RANK/DENSE\_RANK)
5. Combining window functions + aggregations

---------------------------------------------------------------------------------------------------------------------------------Lets Learn 1 by **one:**

Exactly! 😄 You caught that — using a CTE here doesn’t change the **logic** of your query. It’s essentially the same query you already wrote, just wrapped in a temporary “named result set” (AccountTotals).

The main benefits of a CTE are:

1. **Readability** – especially useful when your query has multiple steps.
2. **Reusability** – you can reference the CTE multiple times in the same query without repeating the subquery.
3. **Complex transformations** – easier to break down joins, filters, or window functions step by step.

So for your current simple case, your original query is perfectly fine and even more concise. The CTE is just a “fancier” way, mostly useful in bigger queries.

**Step 1: Inside the CTE (Input / Preparation)**

This part **prepares the data**. You calculate total balances per account type.

WITH a AS (

SELECT account\_type, SUM(balance) AS total\_balance

FROM accounts

GROUP BY account\_type -- Aggregate per account type

)

✅ **Input query:**

* Reads raw accounts table
* Groups by account\_type
* Computes SUM(balance)
* Result: a temporary table a with columns:  
  | account\_type | total\_balance |

**Step 2: Outside the CTE (Output / Use)**

This part **uses the prepared data** to get the top 3 results.

SELECT \*

FROM a

ORDER BY total\_balance DESC -- Sort highest first

LIMIT 3; -- Pick top 3

✅ **Output query:**

* Reads from temporary table a (CTE)
* Sorts by total\_balance
* Selects the **top 3 rows**

Exactly! ✅ You got it perfectly.

* The **CTE is only for preparing data** (input/aggregation).
* Any **JOINs, ORDER BY, LIMIT, or filtering** happen **outside the CTE**, because that’s part of your **final output**.

Think of it like this:

1. **Inside CTE → Input / Prepare data**
   * Aggregate
   * Summarize
   * Clean/filter raw data
2. **Outside CTE → Output / Use the prepared data**
   * Join with other tables
   * Sort, filter, limit
   * Select the final columns

✅ So in a JOIN case, yes — the JOIN is **outside the CTE**, because you are combining the prepared data with other tables to create the output.

Exactly! ✅ You’re right — at its **core**, a **CTE (Common Table Expression)** is:

* Just a **temporary, named result set**
* Created with WITH
* Exists only for the **duration of that query**
* Nothing more, nothing extra — no hidden logic

**Basic structure**

WITH cte\_name AS (

SELECT column1, column2

FROM table\_name

WHERE condition

GROUP BY column1

)

SELECT \*

FROM cte\_name;

* WITH cte\_name AS (...) → **defines the temporary table**
* SELECT \* FROM cte\_name → **uses it in your output query**

💡 **Important points to remember:**

1. CTE does **not store data permanently**.
2. Any **joins, filters, sorting, or limits** you want in the final output are done **outside the CTE**.
3. Inside the CTE, you usually do **preparation**: aggregation, filtering, or intermediate calculation.
4. **So yes** — **at its simplest**, a CTE = WITH + **temporary query**. Nothing extra is required.
5. If you want, I can show a **super minimal example with input → CTE → output**, just to make it crystal clear.

Imp-

**1️⃣ When to use a CTE**

* When you want to **break a complex query into smaller, readable parts**.
* When you need to **reuse a subquery** multiple times in a query.
* When you want to **improve readability instead of nesting many subqueries**.

**You can chain multiple CTEs:**

WITH FirstCTE AS (

SELECT \* FROM Orders WHERE amount > 1000

),

SecondCTE AS (

SELECT customer\_id, COUNT(\*) AS order\_count FROM FirstCTE GROUP BY customer\_id

)

SELECT \* FROM SecondCTE WHERE order\_count > 5;