

EDUSHARK TRAINING

Job-Ready Programs for Freshers

Applied Data Analytics & Business Intelligence

A 12-Week Workbook for Beginners

From data foundations to deployed R Shiny dashboards.

Applied Data Analytics & Business Intelligence: A 12-Week Workbook for Beginners

Edition: First, 2026

Publisher: EDUSHARK TRAINING

Web: <https://edusharktraining.com>

Contact: info@edusharktraining.com | +91 8105 683 683

This workbook accompanies the EDUSHARK TRAINING Applied Data Analytics program. It is freely distributed to enrolled and prospective learners. All trademarks are the property of their respective owners.

Typeset with $\LaTeX$ using Latin Modern, Helvetica and tcolorbox.

How to Use This Workbook

Welcome to your 12-week data analytics journey. This workbook is built on a simple promise: you do not need a maths degree, prior coding, or expensive software to start a serious career as a data analyst. You need a plan, the right tools, and consistent practice. We provide the first two; the third is up to you.

Who this is for

This book is written for absolute beginners. You will find no leaps that assume you already know SQL or have programmed before. Every concept is introduced with a real example, a small piece of code, and a worked exercise.

How each chapter is laid out

Each chapter covers one week (seven days) of study. Inside, you will find the following recurring elements:

- **Learning Objectives** (deep blue box): what you will be able to do by the end of the chapter.
- **Concept boxes** (light blue): a definition or core idea you should memorise.
- **Pro Tip** (mint): a small piece of practitioner wisdom.
- **Watch Out** (coral red): a common pitfall.
- **Try It Yourself** (gold): a short exercise. Try it before peeking at the solution at the back of the workbook.
- **End-of-Week Project** (cream card): a small portfolio piece that ties the week together.
- **Key Takeaways** (mint card): one-line summaries you can scan before exams or interviews.

Tools you will install

You will install the following free tools (one at a time, when each is introduced): Microsoft Excel or Google Sheets, MySQL or PostgreSQL, R, RStudio Desktop, optionally Power BI Desktop and Tableau Public. None require a paid licence to follow this book.

Companion materials

The course website hosts:

- Per-week blog posts mirroring each chapter.
- Sample datasets in `/data/` for every week.
- A separate *Exercises & Solutions* workbook with hundreds of practice problems.

How to Use This Workbook

Now turn the page. Twelve weeks from today, you will have a deployed dashboard, a portfolio, and the confidence to interview for analyst roles.

Contents

How to Use This Workbook	i
1 Data Foundations & Excel Analytics	1
1.1 Day 1 – Introduction to Data Analytics	1
1.2 Day 2 – Data Types and Structures	2
1.3 Day 3 – Business Questions & Problem Framing	3
1.4 Day 4 – KPIs & Metrics Design	3
1.5 Day 5 – Excel Fundamentals for Analytics	4
1.6 Day 6 – Advanced Excel & Pivot Tables	5
1.7 Day 7 – Project: Sales Performance Dashboard	6
2 SQL Fundamentals & Database Basics	7
2.1 Day 1 – Why Databases Exist	7
2.2 Day 2 – Your First SELECT	8
2.3 Day 3 – Filtering with WHERE	8
2.4 Day 4 – ORDER BY, GROUP BY, HAVING	9
2.5 Day 5 – Functions That Clean Real Data	9
2.6 Day 6 – JOINS	10
2.7 Day 7 – Project: E-commerce SQL Report	11
3 Advanced SQL & Analytics Queries	13
3.1 Day 1 – Subqueries	13
3.2 Day 2 – Correlated Subqueries	14
3.3 Day 3 – Window Functions, Part 1: Ranking	14
3.4 Day 4 – Window Functions, Part 2: Running & Comparing	15
3.5 Day 5 – Common Table Expressions (CTEs)	15
3.6 Day 6 – Query Optimisation, the Beginner’s Edition	16
3.7 Day 7 – Project: Advanced Analytics Report	17
4 Statistics Foundations for Data Analysis	19
4.1 Day 1 – Central Tendency	19
4.2 Day 2 – Dispersion	20
4.3 Day 3 – Shape: Distributions	21
4.4 Day 4 – Probability Fundamentals	21
4.5 Day 5 – Common Distributions	22
4.6 Day 6 – Sampling & Estimation	22
4.7 Day 7 – Project: Statistical Analysis Report	23
5 Advanced Statistics & EDA	25
5.1 Day 1 – Hypothesis Testing	25
5.2 Day 2 – The Common Tests	26
5.3 Day 3 – Correlation, not Causation	26

5.4	Day 4 – Exploratory Data Analysis	27
5.5	Day 5 – A/B Testing	27
5.6	Day 6 – Visualising for Analysis	28
5.7	Day 7 – Project: EDA & Statistical Analysis	28
6	R Programming Fundamentals	31
6.1	Day 1 – R, RStudio and Hello World	31
6.2	Day 2 – Data Types and Structures	32
6.3	Day 3 – Control Flow & Functions	32
6.4	Day 4 – The Tidyverse and the Pipe	33
6.5	Day 5 – <code>dplyr</code> : Five Verbs Run Half Your Job	34
6.6	Day 6 – Reshaping with <code>tidyr</code> & Joins	34
6.7	Day 7 – Project: Data Wrangling Pipeline	35
7	Advanced R & Data Visualization	37
7.1	Day 1 – <code>ggplot2</code> Fundamentals	37
7.2	Day 2 – Faceting, Themes, Palettes	38
7.3	Day 3 – Strings with <code>stringr</code>	38
7.4	Day 4 – Dates with <code>lubridate</code>	39
7.5	Day 5 – Advanced <code>dplyr</code>	39
7.6	Day 6 – Project Hygiene	40
7.7	Day 7 – Project: R Analytics Report	41
8	R Shiny – Interactive Dashboards	43
8.1	Day 1 – Hello, Shiny	43
8.2	Day 2 – UI Components and Layout	44
8.3	Day 3 – Reactivity Deep Dive	44
8.4	Day 4 – Interactive Visualisations	45
8.5	Day 5 – <code>shinydashboard</code> & <code>bslib</code>	46
8.6	Day 6 – Deployment	47
8.7	Day 7 – Project: Interactive Business Dashboard	47
9	Power BI Fundamentals (Optional Track)	49
9.1	Day 1 – The Power BI Stack	49
9.2	Day 2 – Connecting to Data	50
9.3	Day 3 – Power Query	50
9.4	Day 4 – Data Modelling	51
9.5	Day 5 – Visualisations & Reports	51
9.6	Day 6 – DAX Fundamentals	52
9.7	Day 7 – Project: Power BI Sales Dashboard	53
10	Tableau Essentials (Optional Track)	55
10.1	Day 1 – Tableau, in 10 Minutes	55
10.2	Day 2 – Basic Visualisations	56
10.3	Day 3 – Advanced Visuals	56
10.4	Day 4 – Calculated Fields & Parameters	56
10.5	Day 5 – Dashboards	57
10.6	Day 6 – Publishing & Comparing Tools	57
10.7	Day 7 – Project: Published Tableau Dashboard	58

11 Capstone Project – Part 1	59
11.1 Day 1 – Choose Your Project	59
11.2 Day 2 – Sourcing Data	60
11.3 Day 3 – Data Preparation & SQL Pipeline	60
11.4 Day 4 – Statistical Analysis in R	61
11.5 Day 5 – Shiny App, Part 1 – Skeleton	62
11.6 Day 6 – Shiny App, Part 2 – Polish	62
11.7 Day 7 – Checkpoint Review	63
12 Capstone Completion & Career Prep	65
12.1 Day 1 – Ship It	65
12.2 Day 2 – Documentation & Portfolio	66
12.3 Day 3 – Resume & LinkedIn	66
12.4 Day 4 – Technical Interviews	67
12.5 Day 5 – Behavioural Interviews	68
12.6 Day 6 – Presentation Practice	69
12.7 Day 7 – Graduation	69
A Glossary	71
B Further Resources	75
B.1 Free books worth your time	75
B.2 Online practice	75
B.3 Cheat-sheets and references	75
B.4 Communities	75
B.5 Datasets to keep practising on	76
B.6 Newsletters & podcasts	76
B.7 Where to host your portfolio	76

Data Foundations & Excel Analytics

Week 1

By the end of this week you will be able to:

OBJECTIVES

1. Explain what data analytics is and why companies pay for it.
2. Classify any data sample by structure and by measurement type.
3. Convert a vague business request into a SMART analytical question.
4. Design meaningful KPIs for a department or product line.
5. Use Excel formulas, lookups and Pivot Tables to produce a one-page sales analysis dashboard.

1.1 Day 1 – Introduction to Data Analytics

Day 1 What it is, what it pays, and where it lives.

Data analytics is the discipline of turning raw observations into decisions. A retailer logs every till transaction; an analyst studies those logs and answers: *Which stores beat their target? Which products are about to stock out? Which discount actually increased profit?* The technical toolkit is interesting only because the answers are.

Concept

Data analytics is the process of *collecting, cleaning, transforming, visualising and modelling* data so that humans can make better decisions, faster and with less guesswork.

1.1.1 The four flavours of analytics

Every analytics question reduces to one of four shapes. Knowing which shape you are tackling tells you what tools you need.

Type	Question	Example	Tools
Descriptive	What happened?	Sales fell 15% last quarter.	Excel, SQL
Diagnostic	Why did it happen?	Top SKU went out of stock for 3 weeks.	SQL, EDA
Predictive	What will happen?	Next-quarter forecast: -8% if no action.	R, Python
Prescriptive	What should we do?	Re-order SKU-12 every 14 days.	R, OR

1.1.2 The analytics lifecycle

A real project moves through six stages: **(1) frame the question, (2) collect data, (3) clean and shape it, (4) explore (EDA), (5) model or visualise, (6) communicate.** Beginners often jump straight to step 5; experienced analysts spend most of their time in steps 1 to 3, because those decide whether the rest will be useful.

✓ Pro Tip

If you can't explain the question in one sentence to a non-technical stakeholder, do not start work yet. Define the question first.

1.1.3 Where analysts work

You will find analysts in five large industries that hire most freshers: **BFSI** (banks, insurance), **Healthcare & Pharma**, **Retail & E-commerce**, **IT & Product**, and **Operations / Supply Chain**. Each has its own vocabulary – “churn” for telecom, “shrinkage” for retail, “adverse event” for pharma – but the analytical method is the same.

1.2 Day 2 – Data Types and Structures

Day 2 If you can't classify your data, you can't pick the right tool.

1.2.1 Structure: how is it stored?

- **Structured** – rows and columns with a fixed schema. Excel, SQL tables, CSVs.
- **Semi-structured** – has hierarchy but no fixed schema. JSON, XML, log files.
- **Unstructured** – free-form text, audio, images, PDF reports.

1.2.2 Measurement: what kind of value is it?

Family	Sub-type and definition	Example
Numerical	Discrete (countable integers)	# of orders
Numerical	Continuous (any real value)	temperature, weight
Categorical	Nominal (no order)	city, gender
Categorical	Ordinal (ranked)	rating 1–5, T-shirt size
Date/time	Point or interval in time	2026-05-01 14:23

⚠ Watch Out

Storing an ordinal as a plain text label (“Low”, “Medium”, “High”) means your tools will sort it alphabetically. Always encode order explicitly – either with numeric codes or with R/Pandas *ordered factors*.

1.2.3 The six dimensions of data quality

Accuracy, Completeness, Consistency, Timeliness, Uniqueness, Validity. Memorise the list – almost every messy dataset is broken on at least two of these dimensions. Your first job on any new file is a quick quality audit.

1.3 Day 3 – Business Questions & Problem Framing**Day 3 Vague requests are the analyst’s biggest time-sink.**

“Can you pull some numbers on the campaign?” is not a question. It is a starting point. Your value as an analyst is to turn that into something answerable.

💡 Concept

The **SMART** framework: a good analytical question is **S**pecific, **M**easurable, **A**ction-oriented, **R**elevant, and **T**ime-bound.

Vague: “How is the new campaign doing?”

SMART: “By how many percentage points did the May 2026 Mother’s Day email increase 14-day repeat-purchase rate among customers who opened the email, versus a matched control cohort, and is the lift statistically significant?”

1.3.1 The five-question stakeholder interview

Before writing a single query, ask:

1. What decision will be made with this analysis?
2. Who is the audience and what do they already believe?
3. By when is it needed – and what changes if it slips?
4. What does “good” look like? What would change your mind?
5. What data do we already have, and where does it live?

1.4 Day 4 – KPIs & Metrics Design**Day 4 The numbers your dashboard will live or die by.**

💡 Concept

A **metric** is any quantity you measure. A **KPI** (Key Performance Indicator) is a metric tied to a goal, with a target and an owner. “Page views” is a metric. “Monthly active users $\geq 200,000$ ” is a KPI.

1.4.1 Leading vs lagging

Lagging indicators measure outcomes that have already happened: revenue, churn, profit. **Leading** indicators measure inputs that predict the outcome: trial signups, NPS, sales calls booked. A good dashboard pairs each lagging KPI with one or two leading indicators.

1.4.2 A library of starter KPIs

Domain	KPIs you will be expected to know
Sales	Revenue, conversion rate, AOV (avg. order value), CAC, LTV, win rate
Marketing	CTR, CPC, ROAS, engagement rate, MQL→SQL, attribution
Operations	On-time-in-full, utilisation, cycle time, defect rate
Finance	ROI, gross margin, operating margin, cash burn, runway
Product	DAU/MAU, retention curve, activation rate, NPS

🔧 Try It Yourself

Pick a business you use weekly (a streaming service, a food-delivery app, your favourite e-commerce site). Write down three lagging and three leading KPIs you would put on the founder’s dashboard. Justify each in one line.

1.5 Day 5 – Excel Fundamentals for Analytics**Day 5 The first analytics tool. Underrated, ubiquitous, very fast.**

Treat Excel as the warm-up tool: it teaches you cell references, formulas, and structured data, all of which translate directly to SQL and R.

1.5.1 The essentials, by function family

Family	Functions
Aggregate	SUM, AVERAGE, COUNT, COUNTA, MIN, MAX, MEDIAN, MODE
Logical	IF, AND, OR, NOT, IFS, IFERROR
Text	LEFT, RIGHT, MID, LEN, TRIM, CONCAT, TEXT
Date	TODAY, NOW, DATE, YEAR, MONTH, DAY, EOMONTH, NETWORKDAYS
Reference	VLOOKUP, HLOOKUP, INDEX-MATCH, XLOOKUP

1.5.2 Cell references, the part everyone gets wrong

- A1 – relative; shifts when you copy.

- **\$A\$1** – absolute; never shifts.
- **\$A1** or **A\$1** – mixed; column or row locked.

⚠ Watch Out

Forgetting to lock a lookup range with **\$** is the single most common cause of “my dashboard mysteriously breaks when I copy a formula down.”

1.6 Day 6 – Advanced Excel & Pivot Tables

Day 6 Where Excel earns its keep.

1.6.1 Lookups: the right one

VLOOKUP works but is brittle (column number is positional). **INDEX-MATCH** is the classical replacement: `=INDEX(price_col, MATCH(lookup_value, key_col, 0))`. Since Excel 365, **XLOOKUP** is the modern one-line answer: `=XLOOKUP(lookup, keys, prices, "not found")`.

1.6.2 Conditional aggregates

SUMIFS, **COUNTIFS**, and **AVERAGEIFS** take pairs of (*range, criterion*). They replace 80% of the “please filter and then sum” tasks that people do by hand.

1.6.3 Pivot Tables in 60 seconds

1. Convert your data to a Table (**Ctrl+T**).
2. **Insert** → **PivotTable**.
3. Drag categorical fields into *Rows* or *Columns*; numeric fields into *Values*.
4. Change the aggregation in *Value Field Settings* (sum, average, count, ...).
5. Add Slicers for one-click filtering.

✓ Pro Tip

A Pivot Table is just SQL’s **GROUP BY** with a mouse. Once you can build one in your sleep, week 2 will feel familiar.

1.6.4 The five Excel formulas you’ll re-use forever

Memorise these patterns. They cover most one-page dashboards.

```
1 =XLOOKUP(A2, products[id], products[price], "not found")
2 =SUMIFS(orders[amount], orders[region], $E2, orders[month], F$1)
3 =AVERAGEIFS(orders[amount], orders[status], "completed")
4 =COUNTIFS(orders[customer_id], A2, orders[status], "<>cancelled")
5 =IF(B2 >= AVERAGE($B$2:$B$500) + 1.5*STDEV.S($B$2:$B$500),
6   "high", "normal")
```

Listing 1.1: Five Excel patterns

1.7 Day 7 – Project: Sales Performance Dashboard

🚩 End-of-Week Project

Build a one-page Excel dashboard from `week1-sales-data.csv` (provided) that answers:

1. Which region grew fastest in the last 90 days?
2. Which three products contribute most to total revenue?
3. What is the average order value, and how has it moved month-on-month?
4. Pick five KPIs you defined on Day 4 and display them as “KPI cards” (boxed values with last-period delta).
5. Write three actionable recommendations for the sales head.

Deliverable: a single .xlsx file with a **Data** sheet, a **Pivot** sheet, and a **Dashboard** sheet – the last one printable on one A4 page.

Key Takeaways

- Analytics has four flavours; every question reduces to one of them.
- Always classify your data by structure and measurement before opening any tool.
- SMART beats vague every time.
- Lagging KPIs measure outcomes; leading KPIs predict them. Pair them.
- Excel is your warm-up: master cell references, conditional aggregates, and Pivot Tables.

Up next: The same questions, at 1,000× the scale. Welcome to SQL.

SQL Fundamentals & Database Basics

Week 2

By the end of this week you will:

OBJECTIVES

1. Read an Entity–Relationship diagram and identify primary / foreign keys.
2. Write `SELECT` queries with `WHERE`, `ORDER BY`, `LIMIT`.
3. Group, count, sum and filter aggregates with `GROUP BY` and `HAVING`.
4. Use string, numeric and date functions for cleanup.
5. Combine tables with `INNER`, `LEFT` and `RIGHT JOINS`.
6. Deliver an e-commerce SQL analysis report.

2.1 Day 1 – Why Databases Exist

Day 1 Excel cracks at 100,000 rows; databases don't.

A **database** is structured storage that lets many people read and write at the same time, without losing data, and that can answer questions in milliseconds. The most common kind is a **relational database** (or RDBMS), where data lives in tables of rows and columns linked to each other by **keys**.

💡 Concept

A **primary key** is a column (or set of columns) that uniquely identifies every row in a table. A **foreign key** is a column that references the primary key of another table, creating a relationship.

2.1.1 The five tables every e-commerce store has

- **customers** (one row per person)
- **products** (one row per SKU)
- **orders** (one row per checkout)
- **order_items** (one row per product within an order)
- **categories** (one row per product category)

`orders.customer_id` is a foreign key pointing at `customers.customer_id`. `order_items.order_id` points at `orders.order_id`. Most analytical SQL is just joining these tables back together.

✔ Pro Tip

Install **MySQL Workbench** (or pgAdmin for Postgres). Load the sample database `week2-ecommerce-database.sql` provided in `/data/`. You will use it for every example in this chapter.

2.2 Day 2 – Your First SELECT

Day 2 The most useful 30 characters in tech.

```
1 SELECT first_name, last_name, email      -- columns to return
2 FROM   customers                        -- source table
3 WHERE  country = 'India'                -- row filter
4 ORDER BY signup_date DESC               -- sort
5 LIMIT  10;                             -- only the first 10
```

Listing 2.1: The anatomy of a SELECT

2.2.1 Distinct, aliases, comments

- `SELECT DISTINCT country FROM customers;` returns unique values only.
- `SELECT price * 0.18 AS gst FROM products;` renames a column.
- `-` starts a single-line comment; `/* ... */` a block comment.

2.3 Day 3 – Filtering with WHERE

Day 3 Five operators that handle 95% of filters.

Operator	Example
<code>=, <>, <, >, <=, >=</code>	<code>price > 1000</code>
<code>BETWEEN</code>	<code>order_date BETWEEN '2026-01-01' AND '2026-03-31'</code>
<code>IN</code>	<code>country IN ('India','UAE','Singapore')</code>
<code>LIKE</code>	<code>email LIKE '%@gmail.com'</code>
<code>IS NULL</code>	<code>phone IS NULL</code>

⚠ Watch Out

`phone = NULL` is always false. Use `phone IS NULL`. This is the single most common SQL bug for beginners.

2.4 Day 4 – ORDER BY, GROUP BY, HAVING

Day 4 From rows to insights.

2.4.1 Sorting

`ORDER BY total_amount DESC, customer_id ASC` sorts by total amount descending, ties broken by customer id.

2.4.2 The five aggregates

`COUNT(*)`, `SUM(x)`, `AVG(x)`, `MIN(x)`, `MAX(x)`. They collapse many rows into one.

2.4.3 GROUP BY – the Pivot Table of SQL

```

1 SELECT  region,
2         SUM(amount) AS revenue,
3         COUNT(*)    AS n_orders,
4         AVG(amount) AS avg_order_value
5 FROM    orders
6 WHERE   status = 'completed'
7 GROUP BY region
8 HAVING  SUM(amount) > 100000
9 ORDER BY revenue DESC;
```

Listing 2.2: Revenue by region, only regions over 1 lakh

💡 Concept

WHERE filters rows *before* grouping. **HAVING** filters groups *after* aggregation. If you put an aggregate in **WHERE**, the database will refuse to run the query.

2.5 Day 5 – Functions That Clean Real Data

Day 5 Real-world data is messy. These functions fix it.

Family	Functions
String	UPPER, LOWER, TRIM, LENGTH, SUBSTRING, CONCAT, REPLACE
Number	ROUND, CEIL, FLOOR, ABS, MOD, POWER
Date	NOW(), CURDATE(), DATE_FORMAT, DATEDIFF, DATE_ADD, EXTRACT
Logic	CASE WHEN ... THEN ... ELSE ... END
Null	COALESCE, IFNULL, NULLIF
Convert	CAST(x AS DECIMAL(10,2)), CONVERT(...)

```

1 SELECT customer_id,
2        SUM(amount) AS lifetime_value,
3        CASE
4            WHEN SUM(amount) >= 50000 THEN 'platinum'
5            WHEN SUM(amount) >= 10000 THEN 'gold'
6            WHEN SUM(amount) >= 2000 THEN 'silver'
7            ELSE 'bronze'
8        END AS tier
9 FROM orders
10 GROUP BY customer_id;

```

Listing 2.3: Bucketing customers with CASE WHEN

2.6 Day 6 – JOINS

Day 6 Putting tables back together.

💡 Concept

A **JOIN** combines rows from two tables based on a matching column. **INNER** keeps only matches. **LEFT** keeps all rows from the left table even if there is no match (the right side becomes NULL). **RIGHT** is the mirror image. **FULL OUTER** keeps everything.

```

1 SELECT c.name AS category,
2        SUM(oi.quantity * oi.unit_price) AS revenue
3 FROM order_items oi
4 JOIN products p ON p.product_id = oi.product_id
5 JOIN categories c ON c.category_id = p.category_id
6 JOIN orders o ON o.order_id = oi.order_id
7 WHERE o.order_date BETWEEN '2026-05-01' AND '2026-05-31'
8 GROUP BY c.name
9 ORDER BY revenue DESC;

```

Listing 2.4: Revenue per category in May 2026

🔧 Try It Yourself

Write a single query that returns, for every customer in the `week2-ecommerce-database`: the customer's name, their total number of orders, and the date of their most recent order. Customers with zero orders should still appear. (Hint: `LEFT JOIN` from customers to orders.)

2.7 Day 7 – Project: E-commerce SQL Report

🚩 End-of-Week Project

Using the supplied e-commerce schema, write a `.sql` file containing the following queries, each with a one-line comment explaining its purpose:

1. Top 10 customers by lifetime revenue.
 2. Monthly revenue trend for the last 12 months.
 3. Category-level conversion rate (orders / unique visitors), if visit data is provided.
 4. Customers who placed orders in 2024 but not in 2025 (churn candidates).
 5. A single “executive summary” query that returns 8 KPIs in one row.
- Submit the script plus a one-page memo with three insights and three recommendations.

Key Takeaways

- Relational data lives in tables linked by keys.
- `SELECT ... FROM ... WHERE ... GROUP BY ... HAVING ... ORDER BY ... LIMIT` is the canonical query shape.
- `WHERE` runs before grouping; `HAVING` runs after.
- Always use `IS NULL`, never `= NULL`.
- `INNER` for matches, `LEFT` when you must keep unmatched rows from the left table.

Up next: the queries that get you hired – window functions, CTEs and subqueries.

Advanced SQL & Analytics Queries

Week 3

By the end of this week you will:

OBJECTIVES

1. Write scalar, column and table subqueries (correlated and not).
2. Use ROW_NUMBER, RANK, DENSE_RANK, LAG, LEAD, SUM() OVER() for analytics.
3. Refactor nested queries into readable CTEs (WITH ...).
4. Build recursive CTEs for hierarchical data.
5. Read an EXPLAIN plan and recognise the obvious anti-patterns.

3.1 Day 1 – Subqueries

Day 1 A query inside another query.

💡 Concept

A **subquery** is a SELECT statement embedded inside another statement. It can appear in the SELECT, FROM or WHERE clause and returns one of three shapes: a single value (scalar), a single column (list), or a full table (derived).

```
1  -- (1) Scalar: returns one value
2  SELECT name, price
3  FROM    products
4  WHERE   price > (SELECT AVG(price) FROM products);
5
6  -- (2) Column / list: returns one column, used with IN
7  SELECT *
8  FROM    customers
9  WHERE   customer_id IN (SELECT customer_id FROM orders
10                          WHERE order_date >= '2026-01-01');
11
12 -- (3) Derived table: a full result used in FROM
13 SELECT region, avg_order
14 FROM (
```

```

15 SELECT region, AVG(amount) AS avg_order
16 FROM   orders
17 GROUP BY region
18 ) r
19 WHERE  avg_order > 5000;

```

Listing 3.1: Three shapes of subquery

3.2 Day 2 – Correlated Subqueries

Day 2 One inner query per outer row – powerful but slow.

```

1 SELECT e.employee_id, e.name, e.salary, e.department
2 FROM   employees e
3 WHERE  e.salary > (
4         SELECT AVG(e2.salary)
5         FROM   employees e2
6         WHERE  e2.department = e.department    -- correlation here
7     );

```

Listing 3.2: Each employee versus their own department

Watch Out

A correlated subquery re-executes for every outer row. For datasets above a few hundred thousand rows, rewrite it as a window function (Day 3) or a JOIN with a pre-aggregated CTE – often 10–100× faster.

3.3 Day 3 – Window Functions, Part 1: Ranking

Day 3 Aggregate, but keep every row.

Concept

A **window function** computes an aggregate *over a window of rows related to the current row*, without collapsing the result like GROUP BY does. The window is defined by an OVER clause.

```

1 WITH ranked AS (
2     SELECT category_id,
3            product_id,
4            revenue,
5            DENSE_RANK() OVER (
6                PARTITION BY category_id
7                ORDER BY    revenue DESC
8            ) AS rnk
9     FROM   product_revenue
10 )

```

```
11 | SELECT * FROM ranked WHERE rnk <= 3;
```

Listing 3.3: Top 3 products per category

Three ranking functions, three behaviours when there is a tie:

- ROW_NUMBER() – 1, 2, 3, 4 (arbitrary tie-break).
- RANK() – 1, 2, 2, 4 (gaps after ties).
- DENSE_RANK() – 1, 2, 2, 3 (no gaps).

3.4 Day 4 – Window Functions, Part 2: Running & Comparing

Day 4 Running totals, moving averages, period-over-period.

```
1 | SELECT month_start,
2 |       revenue,
3 |       SUM(revenue) OVER (ORDER BY month_start) AS running_total,
4 |       LAG(revenue, 1) OVER (ORDER BY month_start) AS prev_month,
5 |       revenue - LAG(revenue, 1) OVER (ORDER BY month_start)
6 |       AS mom_delta,
7 |       AVG(revenue) OVER (ORDER BY month_start
8 |       ROWS BETWEEN 2 PRECEDING AND CURRENT ROW)
9 |       AS rolling_3m_avg
10 | FROM monthly_revenue;
```

Listing 3.4: Running revenue and month-on-month growth

✓ Pro Tip

ROWS BETWEEN ... PRECEDING AND CURRENT ROW is the frame specification. It defines exactly which rows the window covers. Always be explicit – the default frame differs between databases.

3.5 Day 5 – Common Table Expressions (CTEs)

Day 5 Readable, reusable, debuggable.

💡 Concept

A **CTE** (Common Table Expression) is a named, temporary result set declared with WITH and used in the query that follows. It does not persist beyond the statement.

```
1 | WITH monthly AS (
2 |   SELECT DATE_FORMAT(order_date, '%Y-%m-01') AS month_start,
3 |         SUM(amount) AS revenue
4 |   FROM   orders
5 |  WHERE  status = 'completed'
```

```

6      GROUP BY month_start
7  ),
8  growth AS (
9      SELECT month_start,
10             revenue,
11             LAG(revenue) OVER (ORDER BY month_start) AS prev_rev,
12             (revenue - LAG(revenue) OVER (ORDER BY month_start))
13             / NULLIF(LAG(revenue) OVER (ORDER BY month_start),0)
14             AS mom_pct
15      FROM    monthly
16  )
17  SELECT * FROM growth WHERE mom_pct < 0;

```

Listing 3.5: Chained CTEs – one step per CTE

3.5.1 Recursive CTEs

For hierarchies (org chart, category tree, bill-of-materials):

```

1  WITH RECURSIVE chain AS (
2      SELECT employee_id, manager_id, name, 1 AS lvl
3      FROM    employees
4      WHERE   manager_id IS NULL           -- root
5      UNION ALL
6      SELECT e.employee_id, e.manager_id, e.name, c.lvl + 1
7      FROM    employees e
8      JOIN    chain c ON e.manager_id = c.employee_id
9  )
10 SELECT * FROM chain ORDER BY lvl, name;

```

Listing 3.6: Walking a manager chain

3.6 Day 6 – Query Optimisation, the Beginner’s Edition

Day 6 If your query takes more than five seconds, read this.

3.6.1 The five anti-patterns

1. **SELECT *** on a 50-column table when you need three.
2. Functions on indexed columns (**WHERE YEAR(order_date)=2026** defeats the index; use a date range instead).
3. Correlated subqueries on large tables.
4. **OR** on different columns where each could be indexed separately – often rewriteable as **UNION ALL**.
5. Joining on a string when an integer key exists.

3.6.2 Reading EXPLAIN

Prepend **EXPLAIN** (MySQL) or **EXPLAIN ANALYZE** (PostgreSQL) to any query. Look for:

- `type=ALL` or `Seq Scan` – full table scan.
- `rows` much larger than what your final result returns.
- Using `filesort` on large data – consider an index.

Try It Yourself

On the e-commerce database, write a single query that returns – for each month of 2026 – (a) the total revenue, (b) the same month’s revenue one year earlier, and (c) the year-over-year percentage change. Use a CTE for the monthly aggregation and `LAG()` with a 12-row offset for the comparison.

3.7 Day 7 – Project: Advanced Analytics Report

End-of-Week Project

Using a retail dataset of at least 100,000 rows, deliver a single `.sql` script that:

1. Segments customers into RFM (Recency, Frequency, Monetary) quintiles using window functions.
2. Builds a year-over-year revenue comparison with a single chained-CTE query.
3. Returns the top-3 products per category per month.
4. Identifies stockouts using `LAG` on daily inventory.
5. Includes `EXPLAIN` output for at least two queries, with before/after timings.

Key Takeaways

- Subqueries answer “compared to what?”.
- Window functions aggregate without losing rows – the analyst’s super-power.
- CTEs make complex SQL readable and debuggable; build queries step-by-step.
- Recursive CTEs handle hierarchies natively.
- Read `EXPLAIN`. Index the columns in your `WHERE` and `JOIN` clauses.

Up next: we leave the database and pick up the foundations that turn an SQL operator into a real analyst – statistics.

Statistics Foundations for Data Analysis

Week 4

By the end of this week you will:

OBJECTIVES

1. Compute and interpret mean, median, mode, variance, standard deviation, percentiles and IQR.
2. Identify skewness and kurtosis from a histogram.
3. Apply the empirical rule and compute Z-scores.
4. Use basic probability rules and Bayes' theorem.
5. Recognise the binomial, Poisson and normal distributions and pick the right one.
6. Understand sampling, the Central Limit Theorem, and confidence intervals.

4.1 Day 1 – Central Tendency

Day 1 Where is the middle of your data?

Three measures, three personalities.

Measure	Definition	Robust to outliers?
Mean	$\bar{x} = \frac{1}{n} \sum x_i$	No
Median	The 50th percentile	Yes
Mode	The most frequent value	N/A

💡 Concept

Use the **median** for skewed data (income, house prices, transaction sizes). Use the **mean** for roughly symmetric data (heights, exam scores). When mean $\gg$ median, your distribution has a long right tail.

4.1.1 Variants worth knowing

- **Weighted mean:** each value has a weight, $\bar{x}_w = \frac{\sum w_i x_i}{\sum w_i}$. Used in GPAs, portfolio returns.

- **Trimmed mean:** drop the top/bottom 5%, then average. Compromise between mean and median.

4.2 Day 2 – Dispersion

Day 2 How spread out is your data?

$$\text{Variance: } s^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2$$

$$\text{Standard deviation: } s = \sqrt{s^2}$$

$$\text{Coefficient of variation: } CV = \frac{s}{\bar{x}}$$

Compute these in three tools – they all give the same answer:

```

1 # --- R ---
2 mean(salaries); median(salaries); sd(salaries); var(salaries)
3 quantile(salaries, c(0.25, 0.5, 0.75))
4 IQR(salaries)
5 summary(salaries) # all-in-one
6
7 # --- Excel ---
8 # =AVERAGE(A2:A500)      =MEDIAN(A2:A500)      =STDEV.S(A2:A500)
9 # =VAR.S(A2:A500)         =QUARTILE.INC(A2:A500, 1)
10 # =QUARTILE.INC(A2:A500, 3) - QUARTILE.INC(A2:A500, 1) # IQR
11
12 # --- SQL (PostgreSQL) ---
13 # SELECT AVG(salary), STDDEV_SAMP(salary), VAR_SAMP(salary),
14 #          PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY salary),
15 #          PERCENTILE_CONT(0.50) WITHIN GROUP (ORDER BY salary),
16 #          PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY salary)
17 # FROM employees;
```

Listing 4.1: Descriptive statistics in R, Excel and SQL

✓ Pro Tip

Use the *sample* variance (denominator $n - 1$) almost always. The n -denominator version is only correct when you really do have the whole population.

4.2.1 The five-number summary

Min, Q1, median, Q3, max. Plotted as a box-and-whisker, this single chart tells you centre, spread, skew, and outliers at a glance.

4.2.2 Detecting outliers with IQR

- $IQR = Q_3 - Q_1$.
- Outlier fence: below $Q_1 - 1.5 \cdot IQR$ or above $Q_3 + 1.5 \cdot IQR$.

4.3 Day 3 – Shape: Distributions

Day 3 What does the data “look like”?

4.3.1 Skewness and kurtosis

Skewness measures asymmetry. Positive skew = long right tail (income, delivery times). Negative skew = long left tail (exam scores when most pass). **Kurtosis** measures the heaviness of the tails. High kurtosis means more frequent extremes than a normal distribution predicts – a fact that has broken many a risk model.

4.3.2 The normal distribution

$X \sim N(\mu, \sigma^2)$ with bell-shaped PDF $f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$.

Concept

Empirical Rule (68-95-99.7). For any normal distribution, approximately 68% of values lie within $\pm 1\sigma$ of the mean, 95% within $\pm 2\sigma$, and 99.7% within $\pm 3\sigma$. This is the foundation of every quality-control chart you will ever see.

4.3.3 Z-scores: comparing apples to apples

$z = \frac{x-\mu}{\sigma}$. A z-score expresses how many standard deviations a value is from the mean. A new customer with $z=2.3$ on spending is in the top 1% of customers.

4.4 Day 4 – Probability Fundamentals

Day 4 The maths of uncertainty in two pages.

4.4.1 Three rules to memorise

$$\text{Addition: } P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

$$\text{Multiplication: } P(A \cap B) = P(A) \cdot P(B | A)$$

$$\text{Bayes: } P(A | B) = \frac{P(B | A) P(A)}{P(B)}$$

Try It Yourself

A diagnostic test is 95% accurate. The disease occurs in 1% of the population. A random person tests positive. What is the probability they actually have the disease? (Answer: about 16%. Bayes is the only way to see this. Walk through it on paper.)

4.5 Day 5 – Common Distributions

Day 5 Pick the right one and a lot of analysis writes itself.

Distribution	Type	When to use
Bernoulli	Discrete	A single yes/no trial (coin flip, click).
Binomial	Discrete	# successes in n independent yes/no trials.
Poisson	Discrete	Counts of rare events in fixed time/space.
Geometric	Discrete	Trials until the first success.
Uniform	Continuous	All values equally likely.
Normal	Continuous	Heights, errors, averages of many things.
Exponential	Continuous	Waiting times for Poisson events.

4.6 Day 6 – Sampling & Estimation

Day 6 Why a 1,000-person survey can speak for 1 billion.

💡 Concept

Central Limit Theorem (CLT). The distribution of sample means approaches a normal distribution as the sample size grows, *regardless of the shape of the underlying population*. For $n \geq 30$ the approximation is usually good enough.

4.6.1 Standard error

The standard deviation of the sample mean: $SE = \frac{s}{\sqrt{n}}$. Larger samples = smaller SE = tighter estimate.

4.6.2 Confidence intervals

A 95% CI for the mean is approximately $\bar{x} \pm 1.96 \cdot SE$. *Interpretation:* if we repeated the sampling many times, 95% of such intervals would contain the true population mean.

⚠️ Watch Out

A 95% CI does *not* say “there is a 95% probability that the true mean is in this interval” – that is a Bayesian statement, not a frequentist one. Get this right in interviews.

4.6.3 Sampling methods

Simple random, **stratified** (force proportional representation per stratum), **cluster** (sample groups, then survey inside), **systematic** (every k -th unit). Convenience samples are not statistical samples.

4.7 Day 7 – Project: Statistical Analysis Report

🚩 End-of-Week Project

Using `week4-employee-salary.csv` (provided), deliver a one-page report that:

1. Reports mean, median, mode, SD, IQR for salary, overall and by department.
2. Identifies outliers using both the IQR rule and z-scores; cross-check.
3. Visualises the distribution and labels skewness and kurtosis.
4. Gives a 95% confidence interval for the mean salary in each department.
5. One paragraph: what would you ask the HR head, given these numbers?

Key Takeaways

- Mean for symmetric, median for skewed.
- Variance and SD are the building blocks of every statistical test.
- Empirical rule: 68 / 95 / 99.7 for normal data.
- Bayes flips the conditional probability; learn it cold.
- CLT is why samples work; SE tells you how well.

Up next: from describing data to *testing claims* – hypothesis testing, correlation, and exploratory data analysis.

Advanced Statistics & EDA

Week 5

By the end of this week you will:

OBJECTIVES

1. Formulate null and alternative hypotheses for a business problem.
2. Pick and run the right test (Z, t, χ^2 , ANOVA).
3. Interpret a p-value correctly, including its limits.
4. Distinguish correlation from causation.
5. Perform end-to-end exploratory data analysis on a real dataset.
6. Design an A/B test, size it, and analyse the results.

5.1 Day 1 – Hypothesis Testing

Day 1 Turning a question into a verdict.

 Concept

A **hypothesis test** answers: “Is the difference I see in the data *real*, or could it have happened by chance?”. It does so by assuming a **null hypothesis** (no difference) and asking how surprising the data would be if the null were true. The smaller the probability of seeing data this extreme under the null (the **p-value**), the more we doubt the null.

5.1.1 The five-step ritual

1. State H_0 (null) and H_1 (alternative).
2. Pick a significance level α , usually 0.05.
3. Compute the test statistic and the corresponding p-value.
4. If $p < \alpha$, reject H_0 . Otherwise, fail to reject.
5. Report the result *with a confidence interval*, not just a p-value.

5.1.2 Two errors, two costs

Type I (α): convicting an innocent null – you declared an effect that wasn’t real. **Type II** (β): acquitting a guilty null – you missed a real effect. **Power** = $1 - \beta$.

⚠ Watch Out

$p = 0.04$ does *not* mean “there is a 4% chance the null is true.” It means “if the null were true, we’d see data this extreme 4% of the time.” The distinction matters in interviews and in court.

5.2 Day 2 – The Common Tests

Day 2 Pick the right tool for the data.

Test	Use it when ...
One-sample t	Compare a sample mean against a known value.
Two-sample t	Compare two independent group means.
Paired t	Same subjects before & after.
χ^2 (chi-square)	Two categorical variables – are they independent?
ANOVA (one-way)	Compare three or more group means.
Z-test	Like t-test but population SD known, large n .

```

1 # Are the two groups' mean order values different?
2 t.test(amount ~ group, data = orders_ab, var.equal = FALSE)
3 # Welch's t is the default and safer than Student's

```

Listing 5.1: Two-sample t-test in R

5.3 Day 3 – Correlation, not Causation

Day 3 Just because two things move together ...

💡 Concept

Pearson’s r measures the *linear* association between two numeric variables, ranging from -1 to $+1$. **Spearman’s ρ** is its rank-based cousin; it captures any *monotonic* relationship and is robust to outliers.

5.3.1 A tiny field guide

- $|r| < 0.3$: weak. 0.3 – 0.7 : moderate. > 0.7 : strong.
- $r = 0$ does not mean “no relationship”; only “no *linear* relationship”. Always look at the scatter plot.

- A high correlation between A and B can be caused by a third variable C (*confounder*). Ice-cream sales correlate with drowning. Both depend on summer.

5.4 Day 4 – Exploratory Data Analysis

Day 4 The first hour with every new dataset.

EDA is the structured habit of *looking before you model*. There is no single recipe, but the following routine works on almost any tabular dataset.

1. **Shape:** rows, columns, types. `glimpse(df)` or `df.info()`.
2. **Missingness:** where, how much, why.
3. **Univariate:** histogram and boxplot of every numeric column; bar chart of every categorical column.
4. **Bivariate:** scatter for numeric–numeric; box plot for numeric–categorical; cross-tab + heatmap for categorical pairs.
5. **Correlation matrix** of the numeric columns.
6. **Outliers:** list the top/bottom 0.5% on every numeric variable; investigate the suspicious ones.
7. **Initial questions:** write three hypotheses to test next.

✓ Pro Tip

A useful rule: never go past the *shape-and-missingness* step without writing one sentence describing the dataset in your own words. If you can't, you are not ready to model it.

5.5 Day 5 – A/B Testing

Day 5 Hypothesis testing meets the business.

5.5.1 The setup

You have two versions of something – a webpage, an email, a button colour. You randomly assign each visitor to A (control) or B (treatment) and measure a conversion metric.

5.5.2 Sample size

You need to know in advance how many users to test before stopping. A simple rule of thumb for a 5% absolute lift at 80% power and $\alpha = 0.05$ is roughly $n = 16 \cdot p(1 - p)/d^2$ per arm, where p is baseline rate and d is the absolute detectable lift. Online calculators do this for you; use Evan Miller's or your statistics library.

5.5.3 Common mistakes

- Peeking at results early and stopping when the p-value dips below 0.05 – this is *p-hacking* and inflates Type I error.

- Running for too short a period and missing weekly cycles.
- Ignoring novelty effects – treatment lifts often decay.
- Sample-ratio mismatch: if A got 51% and B got 49%, your random assignment broke. Investigate before reporting.

5.6 Day 6 – Visualising for Analysis

Day 6 Charts that earn their pixels.

Goal	Chart
Distribution of one numeric var.	Histogram, density plot
Comparison of two distributions	Overlaid density, box plots
Trend over time	Line chart
Composition / parts of a whole	Stacked bar (avoid pie for > 3 slices)
Relationship of two numbers	Scatter plot with trend line
Geographic	Choropleth, point map

Watch Out

A few habits to drop:

- Truncated y-axes that exaggerate change.
- Pie charts with more than three slices.
- Rainbow colour scales for ordered data; use a sequential palette.
- 3D anything.

Try It Yourself

You ran an A/B test with 1,200 users per arm. Arm A converted 84 users, arm B converted 117. Compute: (1) the conversion rate for each arm, (2) the absolute and relative lift, (3) a two-proportion z-test statistic and its p-value. Decide whether to ship – and write the one sentence you would put in the experiment summary.

5.7 Day 7 – Project: EDA & Statistical Analysis

End-of-Week Project

Using `week5-marketing-ab-test.csv` (provided):

1. Run a full EDA: shape, missingness, univariate, bivariate plots.
2. State a hypothesis for the campaign's effect on conversion rate.
3. Run the appropriate test (two-proportion z-test or chi-square).

4. Report the lift, the p-value, and a 95% CI for the lift.
5. Write a one-page executive memo with a clear ship/hold recommendation.

Key Takeaways

- Hypothesis testing answers “is this real or random?”.
- Pick the right test based on data shape; report CI alongside p-value.
- Correlation $\neq$ causation. Investigate confounders.
- EDA is a discipline, not a slogan. Follow a routine.
- Run A/B tests to a pre-committed plan; don’t peek early.

Up next: a programming language that lets you do all of this – and build interactive dashboards – in one place.

R Programming Fundamentals

Week 6

By the end of this week you will:

1. Install R and RStudio and write your first script.
2. Use R's core data types: vector, list, matrix, data frame, factor.
3. Write functions, conditionals and use the `apply` family.
4. Read data with `readr` and shape it with the `tidyverse` pipeline.
5. Apply `dplyr` verbs: `select`, `filter`, `mutate`, `arrange`, `summarise`, `group_by`.
6. Reshape data with `tidyr` and join multiple tables.

OBJECTIVES

6.1 Day 1 – R, RStudio and Hello World

Day 1 Setting up the tool you will use for the next six chapters.

Download R from cran.r-project.org, then RStudio Desktop (the free edition) from posit.co. Launch RStudio; the four panes are **Script** (top-left), **Console** (bottom-left), **Environment** / **History** (top-right), **Files** / **Plots** / **Help** (bottom-right).

```
1 # Arithmetic
2 2 + 2                # [1] 4
3 log(100, base = 10)  # [1] 2
4
5 # Variables
6 revenue <- 1250000
7 cost    <- 870000
8 profit  <- revenue - cost
9 profit                # [1] 380000
10
11 # Getting help
12 ?mean
13 help.search("regression")
```

Listing 6.1: A first R session

 Pro Tip

`<-` is R's preferred assignment operator (`Alt+-` in RStudio inserts it). `=` also works but is reserved by convention for function arguments.

6.2 Day 2 – Data Types and Structures

Day 2 The five shapes you'll see every day.

6.2.1 Atomic types

numeric, integer, character, logical, complex, raw. Check with `typeof(x)` or `class(x)`.

6.2.2 Vectors – the workhorse

```
1 prices <- c(199, 249, 89, 1299, 349)
2 in_sale <- prices < 300                # logical vector
3 prices[in_sale]                        # subset
4 mean(prices); sd(prices); range(prices)
```

6.2.3 Data frames – the table

```
1 sales <- data.frame(
2   product = c("A", "B", "C"),
3   units   = c(10, 25, 7),
4   price   = c(199, 249, 89)
5 )
6 sales$revenue <- sales$units * sales$price
7 sales[sales$revenue > 1000, ]
```

6.2.4 Lists and factors

A **list** can hold elements of different types and lengths – the JSON of R. A **factor** stores a categorical variable with a fixed set of levels and (optionally) an order; always use it for ordered categories.

6.3 Day 3 – Control Flow & Functions

Day 3 Make your code re-usable in 90 seconds.

```
1 # Conditional
2 classify <- function(score) {
3   if (score >= 80) "A"
4   else if (score >= 60) "B"
5   else "C"
6 }
```



```

7
8 # Vectorised conditional -- usually preferred
9 ifelse(scores >= 60, "pass", "fail")
10
11 # For loop (rarely needed)
12 total <- 0
13 for (x in 1:10) total <- total + x
14
15 # apply family -- the R way
16 apply(prices, function(p) p * 1.18) # add 18% GST

```

Listing 6.2: Conditionals, loops and functions

 Pro Tip

If you find yourself writing a `for` loop over rows of a data frame, stop. There is almost always a vectorised `dplyr` or `apply` version that is shorter and 10× faster.

6.4 Day 4 – The Tidyverse and the Pipe

Day 4 One philosophy, eight packages, infinite analyses.

Install once: `install.packages("tidyverse")`. Load: `library(tidyverse)`.

 Concept

The **pipe** `%>%` (or the newer `|>`) takes the result on its left and passes it as the *first* argument of the function on its right. It turns nested function calls into a readable left-to-right recipe.

```

1 # Nested -- read inside-out
2 head(arrange(filter(mtcars, hp > 100), desc(mpg)), 5)
3
4 # Piped -- read top-down
5 mtcars %>%
6   filter(hp > 100) %>%
7   arrange(desc(mpg)) %>%
8   head(5)

```

Listing 6.3: Compare: nested vs piped

6.4.1 Reading data

```

1 library(readr)
2 orders <- read_csv("data/orders.csv")
3 write_csv(orders, "data/clean_orders.csv")

```

6.5 Day 5 – dplyr: Five Verbs Run Half Your Job

Day 5 select, filter, mutate, arrange, summarise.

```

1 library(dplyr); library(lubridate)
2
3 orders %>%
4   filter(order_date >= today() - days(90)) %>%
5   inner_join(products, by = "product_id") %>%
6   mutate(revenue = quantity * unit_price) %>%
7   group_by(category) %>%
8   summarise(
9     n_orders = n(),
10    revenue = sum(revenue),
11    avg_aov = mean(revenue)
12  ) %>%
13  arrange(desc(revenue))

```

Listing 6.4: Revenue by category in the last 90 days

Verb	Does
<code>select()</code>	keep / drop columns
<code>filter()</code>	keep rows matching a condition
<code>mutate()</code>	create or modify columns
<code>arrange()</code>	sort rows
<code>summarise()</code>	collapse a group into one row
<code>group_by()</code>	set the grouping for subsequent verbs

6.6 Day 6 – Reshaping with tidyr & Joins

Day 6 Long, wide, and joined.

💡 Concept

Tidy data: one row per observation, one column per variable, one table per kind of observational unit. Almost every painful analysis problem becomes easy once the data is tidy.

```

1 # Wide → long
2 sales_long <- sales_wide %>%
3   pivot_longer(cols = jan:dec, names_to = "month", values_to = "revenue")
4
5 # Long → wide
6 sales_wide_again <- sales_long %>%
7   pivot_wider(names_from = month, values_from = revenue)
8
9 # separate & unite

```

```

10 df %>% separate(name, into = c("first", "last"), sep = " ")
11 df %>% unite("date", year, month, day, sep = "-")

```

Listing 6.5: *pivot longer / wider*

6.6.1 Joins

The dplyr joins map directly to SQL: `inner_join()`, `left_join()`, `right_join()`, `full_join()`, `semi_join()` (keep matching rows), `anti_join()` (keep non-matching).

```

1 orders %>%
2   left_join(customers, by = "customer_id") %>%
3   left_join(products, by = "product_id")

```

Try It Yourself

Load the built-in dataset `mtcars`. Using one piped dplyr chain, return – per cylinder count – the count of cars, the mean mpg, and the heaviest car’s model name. Hint: convert the rownames to a column first with `tibble::rownames_to_column()`.

6.7 Day 7 – Project: Data Wrangling Pipeline

End-of-Week Project

Build a single R script `wrangle.R` that:

1. Reads four CSV files (orders, items, customers, products) from `/data/week6/`.
2. Joins them into one tidy table.
3. Cleans column names with `janitor::clean_names()`.
4. Handles missing prices by imputing the category median.
5. Pivots to a monthly revenue table (one row per category, one column per month).
6. Writes the final tidy file to `data/analysis_ready.csv`.

Key Takeaways

- Vectorise. Loops are rarely the answer.
- The pipe makes data analyses readable as recipes.
- Five dplyr verbs cover most everyday transformations.
- tidyr reshapes; dplyr computes; together they tame real-world data.

Up next: we turn these transformed tables into elegant visualisations – and write production-ready R.

Advanced R & Data Visualization

Week 7

By the end of this week you will:

1. Build publication-quality charts with `ggplot2`.
2. Use facets, themes and custom palettes confidently.
3. Handle messy strings with `stringr` and `regex`.
4. Work with dates and times via `lubridate`.
5. Use advanced `dplyr`: `across()`, `case_when()`, window functions.
6. Write clean R scripts you would be happy to show in an interview.

OBJECTIVES

7.1 Day 1 – ggplot2 Fundamentals

Day 1 The grammar of graphics in one page.

💡 Concept

A `ggplot2` plot is built in *layers*: a dataset, an aesthetic mapping (`aes()`: which column → which visual property), one or more geometric layers (`geom_*`), optional scales, facets, and a theme. Each `+` adds a layer.

```

1 library(ggplot2)
2
3 ggplot(mtcars, aes(x = wt, y = mpg)) + geom_point()           # scatter
4 ggplot(mtcars, aes(x = wt, y = mpg)) + geom_smooth()         # trend
5 ggplot(mpg, aes(x = class)) + geom_bar()                     # bar (count)
6 ggplot(mpg, aes(x = displ)) + geom_histogram(bins = 30)      # histogram
7 ggplot(mpg, aes(x = class, y = hwy)) + geom_boxplot()        # box
8 ggplot(economics, aes(x = date, y = unemploy)) + geom_line() # line
9 ggplot(diamonds, aes(carat, price)) + geom_density2d()       # density

```

Listing 7.1: The seven canonical chart types

7.1.1 Adding titles and labels – always

```

1 | ggplot(orders_by_month, aes(month, revenue)) +
2 |   geom_col(fill = "#0077B6") +
3 |   labs(title = "Monthly revenue, 2026",
4 |        subtitle = "Source: order_items",
5 |        x = NULL, y = "Revenue (INR)",
6 |        caption = "Created with R + ggplot2") +
7 |   theme_minimal(base_size = 12)

```

7.2 Day 2 – Faceting, Themes, Palettes

Day 2 Same chart, many lenses.

```

1 | ggplot(sales, aes(month, revenue, group = region)) +
2 |   geom_line(colour = "#00416A", linewidth = 0.7) +
3 |   facet_wrap(~ region, ncol = 3, scales = "free_y") +
4 |   labs(title = "Revenue trend per region") +
5 |   theme_minimal(base_size = 11) +
6 |   theme(strip.background = element_rect(fill = "#E3F2FD", colour = NA),
7 |         strip.text = element_text(face = "bold"))

```

Listing 7.2: Faceted small multiples

7.2.1 Pick a palette on purpose

- Sequential (Blues) for ordered numeric data.
- Diverging (RdBu) for data with a meaningful zero.
- Qualitative (Set2) for unordered categories.
- viridis is colour-blind friendly and prints well in grayscale.

7.2.2 Saving with ggsave

```

1 | ggsave("plots/revenue_trend.png", plot = last_plot(),
2 |       width = 8, height = 5, dpi = 300)

```

7.3 Day 3 – Strings with stringr

Day 3 Cleaning the world's messy text data.

```

1 | library(stringr)
2 |
3 | str_to_upper("hello")           # "HELLO"
4 | str_length("data analytics")    # 14
5 | str_trim("  whitespace ")      # "whitespace"

```

```

6 str_detect(emails, "@gmail\\.com") # logical vector
7 str_replace_all(phones, "[^0-9]", "")
8 str_extract(text, "[A-Z]{2,}")      # any all-caps token
9 str_split(name, " ", simplify = TRUE)

```

Listing 7.3: A field guide

7.3.1 Regex in 30 seconds

- . – any character. * – zero or more.
- \d – digit. \w – word char. \s – whitespace.
- [abc] – any of a, b, c. [^abc] – none of.
- ^ – start, \$ – end. () – capture group.

✓ Pro Tip

Build regex on regex101.com. It explains every token in plain English and saves hours.

7.4 Day 4 – Dates with *lubridate*

Day 4 Parsing and arithmetic that finally make sense.

```

1 library(lubridate)
2
3 ymd("2026-05-16")           # Date
4 dmy("16/05/2026")          # same
5 ymd_hms("2026-05-16 14:23:11")
6
7 today() + days(7)           # one week from now
8 today() %--% ymd("2026-12-31") / months(1) # months remaining
9
10 orders %>%
11   mutate(
12     month    = floor_date(order_date, "month"),
13     weekday  = wday(order_date, label = TRUE),
14     is_weekend = wday(order_date) %in% c(1, 7)
15   )

```

7.5 Day 5 – Advanced *dplyr*

Day 5 across(), case_when() and friends.

```

1 # Apply same function to many columns
2 df %>% summarise(across(where(is.numeric), mean, na.rm = TRUE))
3
4 # Multi-branch logic
5 orders %>% mutate(

```

```

6   tier = case_when(
7     lifetime_value >= 50000 ~ "platinum",
8     lifetime_value >= 10000 ~ "gold",
9     lifetime_value >= 2000 ~ "silver",
10    TRUE ~ "bronze"
11  )
12 )
13
14 # Window functions in dplyr
15 orders %>%
16   group_by(customer_id) %>%
17   arrange(order_date) %>%
18   mutate(order_rank = row_number(),
19          days_since_prev = as.numeric(order_date - lag(order_date)))

```

Listing 7.4: The patterns you'll use forever

7.6 Day 6 – Project Hygiene

Day 6 Code that won't embarrass you in six months.

7.6.1 A clean project layout

Folder	Contains
data/	raw and processed CSVs (gitignore the raw)
R/	helper functions (utils.R, plots.R)
scripts/	top-level pipelines: 01_clean.R, 02_edat.R, 03_model.R
output/	generated plots and tables
README.md	how to run, in five lines or fewer

7.6.2 R Markdown / Quarto

For analyses meant to be read, write .Rmd or .qmd files that mix prose, code, and output. Render to HTML or PDF with one button.

✓ Pro Tip

Use the `here::here()` function for paths and `renv` for package versions. Future you and your colleagues will thank you.

🔧 Try It Yourself

Using `economics` from `ggplot2`: build a faceted line chart with one panel per decade of the data, showing `unemploy` over time. Use `lubridate` to derive the decade column. Add a horizontal reference line at the overall mean. Save the result to `plots/unemployment.png` at 8x5 inches, 300 dpi.

7.7 Day 7 – Project: R Analytics Report

🚩 End-of-Week Project

Deliver a single `.Rmd` report on `week7-retail-sales.csv`:

1. Three faceted `ggplot2` charts of monthly revenue by region.
2. A correlation heat-map of numeric columns.
3. A clean-text section: extract product brands from a free-text `description` column using regex.
4. A time-series chart with a 7-day rolling mean.
5. A short executive summary and three recommendations.

The Rmd should knit to HTML without warnings.

Key Takeaways

- Every `ggplot2` plot is `data + aes + geom + scale + facet + theme`.
- Always set a title, axis labels and a colour-blind-safe palette.
- `stringr` + regex unlock 90% of text-cleaning tasks.
- `lubridate` makes date arithmetic a one-liner.
- Organise projects the same way every time; future-you is paying.

Up next: the chapter that gets you hired – turning all this into a deployed, interactive R Shiny dashboard.

R Shiny – Interactive Dashboards

Week 8

By the end of this week you will:

1. Build a Shiny app from a single file.
2. Understand reactivity: sources, conductors and endpoints.
3. Use the major input widgets and output renderers.
4. Lay out a dashboard with `shinydashboard` and `bslib`.
5. Embed interactive `plotly` charts and DT tables.
6. Deploy your app to `shinyapps.io`.

OBJECTIVES

8.1 Day 1 – Hello, Shiny

Day 1 Your first interactive app in under 30 lines.

 Concept

A Shiny app has two pieces: a `ui` that describes the page, and a `server` function that reads inputs and writes outputs. Shiny re-runs the server logic automatically whenever an input changes – this is **reactivity**, and it is the whole magic.

```
1 library(shiny); library(ggplot2)
2
3 ui ← fluidPage(
4   titlePanel("Hello Shiny"),
5   sidebarLayout(
6     sidebarPanel(
7       sliderInput("bins", "Number of bins:", 5, 50, 30)
8     ),
9     mainPanel(
10      plotOutput("hist")
11    )
12  )
13 )
```

```

14
15 server <- function(input, output, session) {
16   output$hist <- renderPlot({
17     ggplot(faithful, aes(eruptions)) +
18       geom_histogram(bins = input$bins,
19                     fill = "#0077B6", colour = "white") +
20     theme_minimal()
21   })
22 }
23
24 shinyApp(ui, server)

```

Listing 8.1: A complete one-file Shiny app

Save as `app.R` and click *Run App* in RStudio.

8.2 Day 2 – UI Components and Layout

Day 2 The vocabulary of dashboards.

8.2.1 Inputs

`textInput()`, `numericInput()`, `sliderInput()`, `selectInput()`, `checkboxInput()`, `checkboxGroupInput()`, `radioButtons()`, `dateInput()`, `dateRangeInput()`, `fileInput()`, `actionButton()`.

8.2.2 Outputs

`plotOutput()` – `renderPlot()`.
`tableOutput()` – `renderTable()`.
`DT::DTOutput()` – `DT::renderDT()`.
`textOutput()` – `renderText()`.
`uiOutput()` – `renderUI()` (dynamic UI).

8.2.3 Layouts

`fluidPage()`, `fixedPage()`, `sidebarLayout()`, `fluidRow()` + `column()`, `tabsetPanel()`, `navbarPage()`.

8.3 Day 3 – Reactivity Deep Dive

Day 3 The mental model you must own.

💡 Concept

Reactivity has three roles:

- **Reactive sources** – inputs (`input$x`).
- **Reactive conductors** – `... reactive( )` expressions that depend on sources and feed endpoints.

- **Reactive endpoints** – outputs and observers (`output$plot`, `observe({})`).

Shiny lazily re-evaluates conductors only when their inputs change, and only when something downstream actually needs them.

```

1 server <- function(input, output, session) {
2
3   filtered <- reactive({
4     orders %>%
5       filter(region == input$region,
6             order_date %within% input$dr[1] %--% input$dr[2])
7   })
8
9   output$kpi <- renderText(sum(filtered())$amount)
10  output$chart <- renderPlot(ggplot(filtered(), aes(...)) + ...)
11  output$table <- DT::renderDT(filtered())
12 }

```

Listing 8.2: Cache an expensive filter with `reactive()`

8.3.1 When to use what

- `reactive()`: computes a *value* that other code reads.
- `observe()`: runs for side effects (writes a file, updates a widget).
- `observeEvent(input$btn, {})`: explicit, run on button click.
- `eventReactive(input$btn, {})`: computed value, refreshed only when the button fires.

Watch Out

Calling a reactive without parentheses (`filtered`) returns the function itself, not the value. Always `filtered()`. This is a top-three rookie bug.

8.4 Day 4 – Interactive Visualisations

Day 4 Beyond static plots.

```

1 library(plotly); library(DT)
2
3 ui <- fluidPage(
4   plotlyOutput("scatter"),
5   DTOutput("tbl")
6 )
7
8 server <- function(input, output, session) {
9
10  output$scatter <- renderPlotly({
11    plot_ly(mtcars, x = ~wt, y = ~mpg, type = "scatter",
12          mode = "markers", color = ~factor(cyl)) %>%
13    layout(title = "Weight vs MPG")

```

```

14   })
15
16   output$tbl1 <- renderDT({
17     datatable(mtcars, filter = "top",
18               options = list(pageLength = 10))
19   })
20 }

```

Listing 8.3: *Plotly + DT*

Click-, hover- and brush-events from `plotly` can be read back into the server via `event_data("plotly_click")`, letting you build true drill-down interfaces.

8.5 Day 5 – shinydashboard & bslib

Day 5 Professional-looking UI without writing CSS.

```

1  library(shinydashboard)
2
3  ui <- dashboardPage(
4    dashboardHeader(title = "Sales Cockpit"),
5    dashboardSidebar(
6      sidebarMenu(
7        menuItem("Overview", tabName = "overview", icon = icon("home")),
8        menuItem("Customers", tabName = "customers", icon = icon("user")),
9        menuItem("Products", tabName = "products", icon = icon("box"))
10     )
11  ),
12  dashboardBody(
13    tabItems(
14      tabItem("overview",
15        fluidRow(
16          valueBoxOutput("rev"),
17          valueBoxOutput("orders"),
18          valueBoxOutput("aov")
19        ),
20        fluidRow(box(plotOutput("trend"), width = 12))
21      ),
22      tabItem("customers", DTOutput("cust_tbl")),
23      tabItem("products", plotOutput("prod_chart"))
24    )
25  )
26 )

```

Listing 8.4: *A KPI dashboard skeleton*

8.5.1 Modern themes with bslib

`bs_theme(version = 5, bootswatch = "minty")` drops a polished Bootstrap-5 theme into any Shiny app. Combine with `cards()` and `layout_columns()` from `bslib` for an entirely custom-CSS-free dashboard.

8.6 Day 6 – Deployment

Day 6 From localhost to a public URL.

8.6.1 shinyapps.io in five clicks

1. Create a free account at shinyapps.io.
2. In RStudio: Tools > Global Options > Publishing > Connect.
3. Open your `app.R`, click the blue *Publish* icon.
4. Pick the files to deploy. RStudio installs everything in the cloud.
5. Open the URL it gives you. Done.

8.6.2 What to do before deploying

- Move large data into `.rds` files (much smaller than CSV).
- Pin package versions with `renv` so the cloud build matches your laptop.
- Wrap risky operations in `tryCatch()` so a bad input doesn't crash the whole app.
- Add a `showNotification()` for slow operations.

✓ Pro Tip

The free tier gives you five apps and 25 hours of compute / month. Plenty for a portfolio piece. For heavier loads, deploy with Docker on a VPS.

🔧 Try It Yourself

Extend the Day 1 *Hello Shiny* app: add a `selectInput` to choose between `eruptions` and `waiting` (both columns of the `faithful` dataset), and a `checkboxInput` to toggle a density overlay on the histogram. Wrap the data selection in a `reactive()` so both outputs share one evaluation.

8.7 Day 7 – Project: Interactive Business Dashboard

🚩 End-of-Week Project

Deploy a multi-tab Shiny dashboard for the e-commerce dataset:

1. Tab 1: KPI overview (revenue, AOV, orders, conversion) with date range filter.
2. Tab 2: Region drill-down – click a region to filter the product chart below.
3. Tab 3: Customer table (DT) with column filters and CSV export.
4. Tab 4: Plotly scatter – units vs revenue, sized by margin, coloured by category.
5. Deployed to `shinyapps.io` with a public URL.

Submit: GitHub repo link + live URL.

Key Takeaways

- Shiny = ui + server + reactivity.
- Cache expensive computations in `reactive()`.
- `plotly` and `DT` add interaction with no extra HTML.
- `shinydashboard` / `bslib` make apps look professional, fast.
- Deployment to `shinyapps.io` is five clicks; do it for every portfolio project.

Up next: the BI tools the rest of the industry uses – so you know when to reach for them, and when R Shiny is the better answer.

Power BI Fundamentals (Optional Track)

Week 9

By the end of this week you will:

OBJECTIVES

1. Navigate the three views of Power BI Desktop: Report, Data, Model.
2. Connect to Excel, CSV, SQL and web data sources.
3. Shape data in Power Query with applied steps.
4. Build a clean star-schema data model.
5. Write basic DAX measures, including time intelligence.
6. Publish reports to the Power BI Service.

9.1 Day 1 – The Power BI Stack

Day 1 Three apps, one workflow.

- **Power BI Desktop** – the free Windows authoring tool. This is where you do almost all the work.
- **Power BI Service** – the cloud workspace where you publish, share and schedule refreshes.
- **Power BI Mobile** – read-only apps for iOS/Android.

💡 Concept

Power BI is built on three engines: **Power Query** (for shaping data), the **VertiPaq** in-memory columnar database, and **DAX** (for modelling and measures). Knowing which tool to reach for in each engine is half the skill.

9.1.1 When to use Power BI vs R Shiny

Use Power BI when ...	Use R Shiny when ...
You need fast, standardised business reports for many users.	You need bespoke logic, custom models or non-tabular interactions.
Your stakeholders already live in Microsoft 365.	Your reports are research-driven or need to embed statistical models.
You want row-level security with AAD groups.	You want full control of the UI and code.

9.2 Day 2 – Connecting to Data

Day 2 Get Data is where every project starts.

Home > Get Data brings up a catalogue of 100+ connectors: Excel, CSV, SQL Server, MySQL, PostgreSQL, SharePoint, REST APIs, and hundreds more. Choose your mode at import time:

- **Import** – data is cached in the model. Fastest queries; scheduled refresh required.
- **DirectQuery** – queries run against the source on every interaction. Always live, but slower and limited DAX.
- **Live Connection** – only for Analysis Services / PBI datasets.

9.3 Day 3 – Power Query

Day 3 The data-cleaning workhorse.

Click *Transform Data* to open the **Power Query Editor**. Every action you take here is recorded as a step in the *Applied Steps* pane. This is the M language behind the scenes – but you rarely need to write it by hand.

9.3.1 The ten transformations you'll use every day

1. Promote first row to headers.
2. Change column types.
3. Filter rows; remove blanks and duplicates.
4. Replace values; trim and clean text.
5. Split column by delimiter.
6. Pivot / Unpivot columns.
7. Merge queries (left/inner/anti join).
8. Append queries (union).
9. Add custom column with an M expression.
10. Group by with an aggregation.

✓ Pro Tip

Do as much shaping as possible in Power Query, not in DAX. PQ runs once at refresh; DAX runs on every visual interaction.

9.4 Day 4 – Data Modelling

Day 4 Star schemas are not optional.

💡 Concept

A **star schema** has one or more **fact tables** (orders, sales, events – big, narrow, additive) joined to several **dimension tables** (date, customer, product – small, descriptive). Power BI's engine is optimised for this shape. Wide “flat” tables are an anti-pattern.

9.4.1 Relationships

Drag a column from one table to another in the Model view. Set:

- **Cardinality:** usually *Many to One* – many fact rows per dimension row.
- **Cross-filter direction:** *Single* by default. *Both* is rarely needed and easy to misuse.

9.4.2 A date table – always create one

```

1 DataTable =
2 ADDCOLUMNS(
3     CALENDAR(DATE(2020,1,1), DATE(2030,12,31)),
4     "Year",          YEAR([Date]),
5     "MonthNum",      MONTH([Date]),
6     "MonthName",     FORMAT([Date], "mmm"),
7     "Quarter",       "Q" & FORMAT(QUARTER([Date]), "0"),
8     "YearMonth",     FORMAT([Date], "yyyy-mm")
9 )

```

Listing 9.1: DAX: a calendar table

Mark it as a date table (Modeling > Mark as date table) and relate your fact's date column to it.

9.5 Day 5 – Visualisations & Reports

Day 5 One canvas, many pages.

9.5.1 Picking the right visual

Power BI's visualisation pane has 30+ built-in visuals. The same rules apply as in chapter 5: line for trends, bar for ranks, scatter for relationships. **Avoid** 3D charts, dual-axis gauges and rainbow heatmaps.

9.5.2 Slicers, filters, drill-through

- **Slicer** – a visible filter widget.
- **Filter pane** – finer-grained, off to the side.
- **Drill-through** – click a category in a chart, jump to a detail page filtered to that value.

9.6 Day 6 – DAX Fundamentals

Day 6 The language of business measures.

💡 Concept

DAX (Data Analysis Expressions) is the formula language used to define *calculated columns* (per-row) and *measures* (aggregated on demand). Almost always, you want a measure: it adapts to whatever filter context the visual provides.

```

1 Total Revenue := SUM(Sales[Amount])
2
3 YTD Revenue := TOTALYTD([Total Revenue], DateTable[Date])
4
5 Revenue PY := CALCULATE([Total Revenue],
6 SAMEPERIODLASTYEAR(DateTable[Date]))
7
8 YoY % := DIVIDE([Total Revenue] - [Revenue PY],
9 [Revenue PY])
10
11 Top 5 Customers := CALCULATE(
12 [Total Revenue],
13 TOPN(5, VALUES(Customer[Name]), [Total Revenue])
14 )

```

Listing 9.2: Five DAX measures you'll re-use forever

9.6.1 CALCULATE – the keystone

`CALCULATE(expr, filter1, filter2, ...)` evaluates `expr` in a modified filter context. Mastering `CALCULATE` is the difference between a Power BI beginner and a pro.

🔧 Try It Yourself

You have a measure `[Total Revenue]` on a star-schema model with a proper `DateTable`. Write three DAX measures: (1) Revenue in the last 7 days, (2) Revenue same period last year, (3) Year-over-year % growth. Then put all three on a card visual filtered by current month.

9.7 Day 7 – Project: Power BI Sales Dashboard

🚩 End-of-Week Project

Using the e-commerce dataset:

1. Connect Orders, Items, Customers, Products tables (use the provided .pbix starter).
2. Build a star schema with relationships.
3. Create a date table and time-intelligence measures.
4. Build a three-page report: *Overview, Customers, Products*.
5. Add slicers for region, category and date range.
6. Publish to the Power BI Service; share the link.

Key Takeaways

- Shape in Power Query, model in DAX, visualise in the report canvas.
- Star schema beats one flat table every time.
- Always create a proper date table.
- Measures > calculated columns. Both > visual-level formulas.
- CALCULATE is the spine of advanced DAX.

Up next: the third tool in your BI portfolio – Tableau.

Tableau Essentials (Optional Track)

Week 10

By the end of this week you will:

1. Connect to data and navigate Tableau Desktop.
2. Build the seven canonical chart types in Tableau.
3. Use the Marks card – colour, size, shape, label, detail, tooltip.
4. Create calculated fields, parameters and table calculations.
5. Assemble worksheets into an interactive dashboard with filter actions.
6. Publish to Tableau Public.

OBJECTIVES

10.1 Day 1 – Tableau, in 10 Minutes

Day 1 The vocabulary you must know on Day 1.

Term	Means
Data source	Connection to a file or database.
Worksheet	One view (one chart).
Dashboard	A canvas combining many worksheets.
Story	A sequence of dashboards.
Dimension	A descriptive field (usually categorical).
Measure	A numeric field that you aggregate.
Marks card	The panel that maps fields to visual properties.
Shelves	Rows / Columns / Filters / Pages.

✓ Pro Tip

Tableau Public is free, with the catch that all your published work is publicly visible. For portfolio pieces, this is a feature.

10.2 Day 2 – Basic Visualisations

Day 2 Drag, drop, chart.

The flow is always: drag a dimension to Rows, a measure to Columns (or vice versa), and Tableau picks a sensible default. The **Show Me** panel suggests the chart families compatible with your selection.

Charts in this order:

- Bar / Column – ranks and comparisons.
- Line – trends over time.
- Scatter – relationships of two measures.
- Map – geographic, with auto-detected lat/long.
- Bullet, KPI, Big-Number – summaries.

10.3 Day 3 – Advanced Visuals

Day 3 Beyond the Show Me defaults.

- **Dual-axis**: drop a second measure on the right Y-axis, then right-click “Synchronise Axis”.
- **Combination charts**: one measure as bars, another as line.
- **Treemaps / heat maps**: hierarchical compositional data.
- **Reference lines / bands**: averages, targets, control bands.
- **Trend lines**: linear, polynomial, logarithmic; check R-squared in tooltip.

10.4 Day 4 – Calculated Fields & Parameters

Day 4 Logic without leaving Tableau.

💡 Concept

A **calculated field** is a new column computed from existing ones; you write its formula in a Tableau expression. A **parameter** is a single, user-controlled value that any calculation can read – a perfect way to add “what-if” sliders.

10.4.1 The five formula families

- String: UPPER, TRIM, CONTAINS, SPLIT.
- Date: DATEDIFF, DATETRUNC, TODAY.
- Logical: IF/ELSEIF, CASE WHEN, IIF.
- Aggregate: SUM, AVG, COUNTD.

- Table calc: RUNNING_SUM, WINDOW_AVG, INDEX.

```

1 // Customer tier
2 IF [Total Revenue] >= 50000 THEN "Platinum"
3 ELSEIF [Total Revenue] >= 10000 THEN "Gold"
4 ELSEIF [Total Revenue] >= 2000 THEN "Silver"
5 ELSE "Bronze"
6 END
7
8 // % of total within partition (table calc)
9 SUM([Sales]) / TOTAL(SUM([Sales]))

```

Listing 10.1: Examples of calculated fields

10.5 Day 5 – Dashboards

Day 5 Worksheets in a single, opinionated canvas.

10.5.1 Build flow

1. Each chart goes on its own worksheet first; polish there.
2. File > New Dashboard. Pick a device-aware size.
3. Drag worksheets, filters, and parameters onto the canvas.
4. Use **Containers** (Horizontal / Vertical) for clean alignment.
5. Add **Filter Actions**: clicking a chart filters another.

10.5.2 Five design rules

- One clear question per dashboard; one “hero” chart on top.
- Maximum five colours; reserve red/green for delta.
- White space matters. Crowded dashboards hide insight.
- Tooltips should answer the next likely question.
- Always test in a fresh user’s eyes – ideally on a phone.

10.6 Day 6 – Publishing & Comparing Tools

Day 6 Three tools, one career.

10.6.1 Tableau Public

Server > Tableau Public > Save to Tableau Public. The workbook gets a public URL you can pin to your portfolio.

10.6.2 Tableau vs Power BI vs R Shiny

Tool	Best at
Tableau	Visual exploration, beautiful defaults, story telling.
Power BI	Microsoft 365 ecosystems, enterprise governance, DAX measures.
R Shiny	Custom logic, statistical models, fully bespoke UI.

✓ Pro Tip

The interview question is rarely “which is best?” but “why did you pick this one?”. Knowing the trade-offs cold makes you a credible analyst.

🔧 Try It Yourself

Using Tableau’s built-in *Sample - Superstore* dataset: build a dashboard with (a) a bar chart of sales by Category, (b) a line chart of sales over time, (c) a map of sales by State. Add a filter action so that clicking a Category bar filters both the line chart and the map.

10.7 Day 7 – Project: Published Tableau Dashboard

🚩 End-of-Week Project

Re-create the e-commerce dashboard from chapter 8, but in Tableau:

1. Three worksheets: revenue trend, region map, top-products bar.
2. One dashboard with cross-filtering between all three.
3. Two calculated fields (customer tier, YoY %).
4. One parameter – a “target revenue” that drives a reference line.
5. Published to Tableau Public; URL added to your portfolio.

Write a one-paragraph comparison with your Shiny and Power BI versions.

Key Takeaways

- Dimensions go on rows/columns; measures get aggregated.
- Calculated fields and parameters cover most “custom logic” needs.
- Filter actions are what make Tableau dashboards *feel* interactive.
- Always finish a project with a published, public URL.

Up next: two weeks to put everything together – your end-to-end capstone project and the career skills to land your first role.

Capstone Project – Part 1

Week 11

By the end of this week you will:

1. Choose a domain and scope an analytics project end-to-end.
2. Source real-world data ethically and reproducibly.
3. Build a SQL + R analysis pipeline for the project.
4. Wireframe and prototype an R Shiny dashboard.
5. Present a Week 11 checkpoint review for mentor feedback.

OBJECTIVES

11.1 Day 1 – Choose Your Project

Day 1 The single biggest factor in interview outcomes.

A capstone is not graded on technical complexity alone. Recruiters look for: *a clear question, a defensible method, a polished output, and a useful conclusion*. Pick a project where you have access to data and a real audience – yourself or a friend’s business counts.

💡 Concept

A capstone is *not* a portfolio of unrelated charts. It is a single end-to-end story: one business question → one dataset → one method → one deployed artefact → one decision. Every section of your README, every slide of your presentation, and every tab of your Shiny app must pull in the same direction.

11.1.1 Starter ideas, by domain

Domain	Projects
Retail / E-commerce	Sales forecast, customer segmentation, inventory dashboard.
Finance / BFSI	Stock screener, credit-risk app, expense tracker.
Healthcare / Pharma	Clinical-trial KPI tracker, hospital-resource planner, disease-trend explorer.
Marketing	Campaign attribution, A/B test viewer, content-performance dashboard.
Operations	SLA monitor, defect Pareto, fleet utilisation.

11.1.2 The project proposal template

1. **Problem statement** (one paragraph).
2. **Audience** – who would use this, and what decision would they make?
3. **Data sources** – exactly which datasets, where they come from, licensing.
4. **KPIs** – the three to seven numbers your dashboard must compute.
5. **Deliverables** – a deployed Shiny app, GitHub repo, 2-page memo, 5-minute video walkthrough.
6. **Timeline** – daily milestones for Weeks 11–12.
7. **Risks** – what could go wrong, and your mitigation.

11.2 Day 2 – Sourcing Data

Day 2 Five places real datasets actually live.

- **Kaggle Datasets** – well-documented; pick ones with a clear licence.
- **Government open data** – `data.gov.in`, `data.gov.uk`, `data.gov`.
- **UCI Machine Learning Repository** – classics with good provenance.
- **Public APIs** – World Bank, OpenWeather, Spotify, TwitterX (paid tier).
- **Web scraping** – only when the site's ToS allow and you respect `robots.txt`.

Watch Out

Before importing a dataset, read its licence. If you can't redistribute it, include a clear note in your README and link to the original instead of committing the file.

11.3 Day 3 – Data Preparation & SQL Pipeline

Day 3 From raw files to a queryable database.

11.3.1 A cleaning checklist

- Standardise column names (`snake_case`).
- Parse dates and numbers.
- Handle missing values explicitly (drop? impute? flag?).
- De-duplicate.
- Document every choice in a `NOTES.md`.

11.3.2 Load into SQL

```

1 CREATE TABLE staging_orders (
2   order_id      INT,
3   customer_id   INT,
4   order_date    DATE,
5   amount        DECIMAL(12,2),
6   region        VARCHAR(64),
7   status        VARCHAR(16)
8 );
9
10 LOAD DATA LOCAL INFILE 'data/orders.csv'
11 INTO TABLE staging_orders
12 FIELDS TERMINATED BY ','
13 ENCLOSED BY '"'
14 IGNORE 1 LINES;
15
16 CREATE VIEW v_orders_completed AS
17 SELECT * FROM staging_orders WHERE status = 'completed';

```

Listing 11.1: A reproducible loader

11.3.3 Five analytical views to build first

1. Daily / monthly revenue.
2. Customer lifetime value.
3. Top-N products per category.
4. Cohort retention (sign-up month $\times$ months since).
5. Funnel: visit $\rightarrow$ add-to-cart $\rightarrow$ purchase.

11.4 Day 4 – Statistical Analysis in R

Day 4 Stress-test your conclusions.

Run the EDA recipe from chapter 5 on the cleaned data. For every KPI you plan to show:

- Plot its distribution and check for outliers.
- Check for seasonal and weekday effects.
- Test the most interesting comparison with a t-test or chi-square – and report the confidence interval.
- Document at least one finding that surprised you.

✓ Pro Tip

Interviewers love surprising findings. “Tuesday is our biggest revenue day, not Friday, and here is why” is a far better story than “revenue went up”.

11.5 Day 5 – Shiny App, Part 1 – Skeleton

Day 5 Build the bones before the muscles.

11.5.1 Wireframe first

On paper or in any tool, sketch each tab: header, filter sidebar, KPI strip, primary chart, secondary chart, data table. Get a stakeholder to sign off before writing UI code.

11.5.2 Project layout

```
1 capstone/
2   app.R           # entry point (or split into ui.R / server.R)
3   R/
4     mod_overview.R # one Shiny module per tab
5     mod_customers.R
6     mod_products.R
7     plots.R        # ggplot helpers
8     queries.R       # SQL / dplyr helpers
9   data/
10    raw/            # immutable
11    processed/      # built by scripts/
12  scripts/
13    01_load.R
14    02_clean.R
15    03_export_rds.R
16  www/
17    style.css
18    logo.png
19  renv.lock
```

11.5.3 Modules: the right way to grow a Shiny app

For any app over 150 lines, break each tab into a Shiny module (`moduleServer()` + UI function). Modules namespace their inputs, prevent collisions, and let you re-use the same component multiple times.

11.6 Day 6 – Shiny App, Part 2 – Polish

Day 6 Make it production-grade.

11.6.1 Performance

- Pre-aggregate slow queries into `.rds` files; load once at app start.
- Use `bindCache()` to memoise expensive reactives.
- Profile with `... profvis::profvis( )`.

11.6.2 Error handling and UX

- `validate(need(input$x, "Pick a region"))` to gate plots.
- Wrap risky code in `tryCatch` with a friendly error toast.
- Add a loading spinner with `shinycssloaders::withSpinner()`.
- Provide a `downloadButton()` on every table.

11.6.3 Style

Pick one theme via `bslib::bs_theme()` and never look back. Use a consistent palette, generous white-space, and a clear primary action per page.

Try It Yourself

Before Day 7's checkpoint, do a peer-walkthrough rehearsal: present your capstone in exactly 5 minutes to a friend who is *not* a data analyst. Time yourself. The point of failure is almost always slide 2 – your problem statement is jargon-heavy or the audience does not see why the question matters. Re-write it three different ways before the checkpoint.

11.7 Day 7 – Checkpoint Review

Day 7 Submit, present, get feedback.

End-of-Week Project

Week 11 Checkpoint Deliverables

1. Project proposal (1 page).
2. Cleaned data + load scripts in GitHub repo.
3. Five SQL views with documentation.
4. EDA notebook with at least one surprising insight.
5. Working Shiny prototype (need not be deployed yet).
6. 5-minute walkthrough video (Loom or screen-recorded).

Schedule a 30-minute review with your mentor. Bring questions, not defensiveness.

Key Takeaways

- Scope drives everything; pick a project you can finish, not the most impressive idea.
- Document data sources and cleaning choices; reproducibility is a differentiator.
- Build the Shiny app in modules from day one.
- Hunt for one counter-intuitive insight – that's your story.

Up next: ship it, write the README, polish your resume, and walk into your first analyst interview.

Capstone Completion & Career Prep

Week 12

By the end of this week you will:

1. Deploy your capstone Shiny app to a public URL.
2. Write a professional README and portfolio entry.
3. Optimise your resume and LinkedIn for analyst roles.
4. Practice the technical and behavioural interview rituals.
5. Deliver a polished 15-minute capstone presentation.

OBJECTIVES

12.1 Day 1 – Ship It

Day 1 Deployment day.

12.1.1 Deployment checklist

- All R scripts re-run from a fresh checkout without manual steps.
- `renv` (or a `DESCRIPTION` file) pins package versions.
- Secrets (API keys, DB passwords) are read from environment variables, not committed.
- App opens cleanly on a phone-size screen.
- `showNotification("Loading...")` for any slow operation.
- Footer credits you (name, GitHub URL, LinkedIn).

12.1.2 Deploy in five clicks

Open RStudio, open `app.R`, click the blue Publish icon, pick **shinyapps.io**, choose the files, click Publish. Your app gets a URL like `your-name.shinyapps.io/capstone`.

✓ Pro Tip

If your free tier runs out of compute, you can self-host with `shiny-server` on a \$5 VPS, or push a Docker image to **Render**, **Fly.io** or **Posit Connect Cloud**.

12.2 Day 2 – Documentation & Portfolio

Day 2 If it's not on GitHub, it doesn't exist.

12.2.1 The README that lands interviews

A great repo README has exactly these sections, in order:

1. **Title** + one-sentence pitch.
2. Screenshot or animated GIF of the live app.
3. Live demo link.
4. Problem statement: what business question does this answer?
5. Data sources & licence.
6. Methods (a short paragraph, not a textbook).
7. Key findings – 3 bullet points.
8. How to run locally (5 commands max).
9. Folder structure.
10. License + your contact.

12.2.2 GitHub profile README

Make your profile `github.com/<you>/<you>` repo and put a one-page README there. Include: pinned projects, a short bio, skills, contact links, and what you're learning now.

12.3 Day 3 – Resume & LinkedIn

Day 3 Make the first 10 seconds count.

12.3.1 A one-page resume skeleton you can paste-and-edit

```
1 YOUR NAME
2 Aspiring Data Analyst | SQL . R Shiny . Power BI . Excel
3 City, Country | +91 XXXXXXXXXX | your.email@example.com
4 linkedin.com/in/your-handle | github.com/your-handle
5
6 SUMMARY
7 Data analyst trained at EDUSHARK with hands-on capstone in SQL,
8 R Shiny and Power BI. Built and deployed a live analytics
9 dashboard. Seeking junior analyst role in BFSI / Healthcare.
10
11 SKILLS
12 SQL (MySQL, PostgreSQL); R (tidyverse, ggplot2); R Shiny;
13 Power BI (DAX, Power Query); Tableau; Excel; Python (basic); Git.
14
```

```

15 PROJECTS
16 - Retail Sales Cockpit - R Shiny + SQL - [LIVE URL]
17   120k-row dataset. RFM segmentation. Top decile = 38% of revenue.
18   Deployed via renv on shinyapps.io; load time < 2s.
19 - E-commerce SQL Analysis - PostgreSQL - [GITHUB]
20   15 analytical queries; window functions, CTEs; 8x speedup via
21   index tuning documented with EXPLAIN.
22
23 CERTIFICATIONS
24 EDUSHARK TRAINING - Applied Data Analytics & BI (YYYY-MM)
25
26 EDUCATION
27 Degree, University, YYYY-YYYY

```

Listing 12.1: Plain-text resume template (ATS-friendly)

12.3.2 The fresher data-analyst resume, in one page

1. **Header:** name, role you want, email, phone, GitHub, LinkedIn, location.
2. **Summary** (2 lines): “Data analyst trained at EDUSHARK with hands-on projects in SQL, R Shiny and Power BI. Looking for an analyst role in BFSI / Healthcare / Retail.”
3. **Skills:** SQL, R, Tidyverse, R Shiny, Power BI, Tableau, Excel, Python (basics), Git.
4. **Projects** (the heart of the resume, 3–4 entries). Each: title → tech → outcome with a number.
5. **Certifications** – EDUSHARK, plus any free Coursera/Datacamp.
6. **Education** – last because projects matter more.

Watch Out

ATS killers to avoid: tables (many scanners can’t read them), icons-as-images, two-column PDFs from Canva, photos. Plain one-column ATS-friendly templates from flowcv.com or rxresu.me win.

12.3.3 LinkedIn checklist

- Professional photo, brand-coloured banner.
- Headline: “Aspiring Data Analyst | SQL, R Shiny, Power BI”.
- About: same pitch as the resume summary, two paragraphs.
- Featured: pin the capstone live URL and GitHub.
- Skills + endorsements: add the 10 you used in this course.
- Open to Work badge *on* (recruiter-visible).

12.4 Day 4 – Technical Interviews

Day 4 The drills that matter.

12.4.1 The five buckets

1. **SQL live coding:** window functions, CTEs, second-highest salary, sessionisation.
2. **Statistics:** explain p-value, CLT, when to use median.
3. **Case studies:** “revenue dropped 10% – diagnose.”
4. **Tool comparison:** “why R Shiny over Power BI here?”
5. **Your project:** walk me through the capstone.

12.4.2 A practice plan

- LeetCode SQL – aim for 60 problems across Easy + Medium.
- StrataScratch / DataLeMur for case-style questions.
- Khan Academy stats refresher.
- Pair-mock with a peer once a week.

12.5 Day 5 – Behavioural Interviews

Day 5 The hardest hour you’ll prep for.

💡 Concept

The **STAR** framework: answer behavioural questions in four beats – **S**ituation, **T**ask, **A**ction, **R**esult. The result must be quantified.

12.5.1 Prepare seven stories, total

Cover: leadership, conflict, failure, learning quickly, handling ambiguity, persuading without authority, beating a deadline. One strong story per theme is enough; the same story can answer two questions.

12.5.2 The “tell me about yourself” opener

Three minutes, three beats: *Where I am now* (course, recent projects), *how I got here* (background + motivation), *where I want to go* (the role I’m interviewing for).

12.5.3 Questions you should ask

- What does success look like at six months?
- Walk me through a recent analysis the team ran end-to-end.
- What’s the team’s tooling – SQL, BI, scripting languages?
- How is data quality maintained?

12.6 Day 6 – Presentation Practice

Day 6 15 minutes that decide everything.

12.6.1 Slide outline

1. Title + your name + role you want.
2. The business question (one sentence).
3. The data and method (one slide each, max).
4. Key findings – three slides, one finding each.
5. Live demo of the Shiny app (2–3 minutes).
6. Recommendations – three actionable bullets.
7. Q&A prep slide (hidden) with anticipated questions.

12.6.2 Five delivery tips

- Open with the punchline, not the methodology.
- Rehearse the demo on the actual screen-share setup once.
- One number per slide, big, with context.
- Pause after questions; do not fill silence.
- End with a confident, written next-step.

Try It Yourself

Record a 90-second video answer to “Tell me about yourself.” Watch it back with the sound off the first time – check your body language and eye contact. Then with sound – check pacing, filler words and clarity. Three takes is enough. Save the best one; you will use it as the introduction in your first screening call.

12.7 Day 7 – Graduation

Day 7 You did it.

End-of-Week Project

Final Capstone Presentation

- 15-minute live presentation to a panel.
- Live demo of deployed R Shiny dashboard.
- Q&A with peers and instructors.
- Certificate of completion.

Grading

- Capstone project quality – 40%
- Presentation & communication – 30%
- Documentation & portfolio – 20%
- Program participation – 10%

Key Takeaways

- Shipping is the only thing that counts in a portfolio.
- READMEs and screenshots get more eyes than code does.
- ATS-friendly resumes beat beautiful ones.
- Practice SQL live, statistics out loud, and project narration in front of a mirror.
- Behavioural answers in STAR; quantify every result.

Congratulations. You have completed twelve weeks of intensive training in Data Analytics, SQL, R Programming, R Shiny and BI tools. You have a deployed dashboard, a public portfolio, and – most importantly – the habits of an analyst.

Now go and apply.

Glossary

A/B test

A randomised controlled experiment comparing two variants of something to estimate the causal effect of a change.

Aggregate function

A function that collapses a set of rows into a single value (SUM, AVG, COUNT, MIN, MAX, MEDIAN).

ANOVA

Analysis of Variance: a test for differences in means across three or more groups.

ATS

Applicant Tracking System – software recruiters use to screen CVs.

Bayes' theorem

The rule for updating a probability after observing new evidence: $P(A|B) = P(B|A)P(A)/P(B)$.

BFSI

Banking, Financial Services and Insurance – one of the largest hiring sectors for analysts.

CTE

Common Table Expression – a named, temporary result set in SQL declared via WITH.

CLT

Central Limit Theorem – the distribution of sample means approaches normality regardless of the population shape.

Cardinality

The numeric relationship between two related tables (1:1, 1:many, many:many).

Choropleth

A map shaded by a quantitative value per region.

Confidence interval

A range that, under repeated sampling, would contain the true parameter a specified percentage of the time.

Correlation

A measure of the linear (Pearson) or monotonic (Spearman) association between two variables.

DAX

Data Analysis Expressions – the formula language of Power BI.

Descriptive analytics

Analytics that answer “what happened?”.

Diagnostic analytics

Analytics that answer “why did it happen?”.

Dimension table

A descriptive table joined to a fact table in a star schema.

EDA

Exploratory Data Analysis – the structured habit of understanding a dataset before modelling.

Fact table

A table of measurable events (orders, transactions) joined to dimension tables.

Filter context

In DAX, the set of filters active when a measure is evaluated.

GROUP BY

SQL clause that collapses rows into groups for aggregation.

Hypothesis test

A procedure for deciding whether data is statistically consistent with a null hypothesis.

Imputation

Replacing missing values with a plausible substitute (median, mean, model output).

INDEX-MATCH

The classical Excel two-step lookup that survives column reordering, unlike VLOOKUP.

IQR

Interquartile range, $Q_3 - Q_1$. A robust measure of spread.

Join

An operation combining rows from two tables based on a related column.

KPI

Key Performance Indicator – a metric tied to a goal with a target and an owner.

Lagging indicator

A metric measuring outcomes that have already occurred.

Leading indicator

A metric measuring inputs that predict future outcomes.

Measure (DAX)

An aggregated calculation evaluated at query time in Power BI.

Median

The middle value of an ordered dataset; the 50th percentile.

Normal distribution

A symmetric bell-shaped distribution defined by its mean and standard deviation.

Null

In SQL, a marker for unknown / missing values. Always tested with `IS NULL`.

OVER()

The SQL clause that turns an aggregate into a window function.

Outlier

A data point unusually far from the rest of the distribution.

Pivot table

A two-dimensional summary table in Excel; SQL equivalent: `GROUP BY`.

Power Query

The data-shaping engine in Power BI; uses the M language.

Predictive analytics

Analytics that answer “what will happen?”.

Prescriptive analytics

Analytics that answer “what should we do?”.

Primary key

A column or combination that uniquely identifies a row.

P-value

The probability of observing data at least as extreme as the sample, assuming the null hypothesis is true.

R Shiny

An R framework for building interactive web applications and dashboards.

Reactivity

Shiny’s automatic re-evaluation model: outputs update whenever their input dependencies change.

Regex

Regular expression – a pattern language for matching text.

RFM

Recency, Frequency, Monetary – a classic customer segmentation framework.

Sampling

Selecting a subset of a population to estimate its properties.

SMART

Specific, Measurable, Action-oriented, Relevant, Time-bound – a framework for analytical questions.

Standard deviation

The square root of the variance; spread in the data’s units.

STAR

Situation, Task, Action, Result – the framework for behavioural interview answers.

Star schema

A data-model pattern with fact tables surrounded by dimension tables; the canonical BI shape.

Stringr

A Tidyverse package for consistent string manipulation in R.

Subquery

A SELECT statement embedded inside another SQL statement.

Tibble

A modern R data frame from the Tidyverse with cleaner printing and stricter type behaviour.

Tidy data

One row per observation, one column per variable.

Type I error

Rejecting a true null hypothesis (false positive).

Type II error

Failing to reject a false null hypothesis (false negative).

VLOOKUP

A vertical lookup function in Excel; superseded by XLOOKUP.

Window function

A SQL function that computes across a set of rows related to the current row without collapsing them.

XLOOKUP

The modern Excel lookup function that handles errors, horizontal and vertical lookups in one call.

Z-score

Number of standard deviations a value lies from the mean.

Further Resources

B.1 Free books worth your time

- Hadley Wickham & Garrett Golemund, *R for Data Science* (2nd ed.) – the canonical tidyverse book. r4ds.hadley.nz.
- Hadley Wickham, *Mastering Shiny*. mastering-shiny.org.
- Cole Nussbaumer Knafl, *Storytelling with Data*.
- Joe Celko, *SQL for Smarties* – the classical reference.

B.2 Online practice

- leetcode.com/problemset/database – SQL drills.
- stratascratch.com – analytics case-style SQL.
- datalemur.com – SQL interview questions.
- kaggle.com/learn – short, well-paced micro-courses.

B.3 Cheat-sheets and references

- Posit / RStudio Cheatsheets – posit.co/resources/cheatsheets/.
- Microsoft's DAX guide – dax.guide.
- Power BI documentation – learn.microsoft.com/power-bi.
- Tableau Help – help.tableau.com.

B.4 Communities

- stackoverflow.com (tags: `r`, `sql`, `shiny`, `powerbi`).
- RStudio Community – community.rstudio.com.
- `r/dataisbeautiful`, `r/dataengineering` on Reddit.
- Tableau Public gallery – inspiration and recipes.

B.5 Datasets to keep practising on

- Kaggle Datasets (kaggle.com/datasets).
- UCI ML Repository.
- Indian government open data – data.gov.in.
- World Bank Open Data – data.worldbank.org.
- Our World in Data – ourworldindata.org.
- NYC Open Data – opendata.cityofnewyork.us.

B.6 Newsletters & podcasts

- “Data Elixir” weekly newsletter.
- “The Analytics Engineering Roundup”.
- “Towards Data Science” on Medium (filter for quality).
- Podcasts: *Data Skeptic*, *Super Data Science*.

B.7 Where to host your portfolio

- github.io – the free option, infinite control.
- shinyapps.io – free tier for live R Shiny apps.
- tableau.com/public – Tableau dashboards.
- public.powerbi.com – shareable Power BI reports.
- quarto.org/docs/publishing – multi-format publishing.

Bookmark a few, ignore the rest, and ship something this week.