60-Day Beginner-Friendly Training Schedule • With Examples & Quick Reference

This document expands the 60-day curriculum with beginner-friendly explanations and copy-pasteable examples. Each section lists the purpose, mandatory commands/concepts, and short examples. Use this as a quick reference during practice.

Day 1: Unix Commands — Basics I

Goals: Navigate the filesystem, inspect files, and perform basic CRUD on directories/files.

Mandatory commands & concepts:

- **Navigation:** pwd, cd, ls, tree (if installed)
- **Files & Directories:** touch, mkdir, rmdir, cp, mv, rm (-i, -r), echo, cat
- **Hidden files & globs:** ls -a, ls *.log

Examples:

```
# where am I?
pwd

# list (detailed/hidden/sort by time)
ls -l
ls -la
ls -lt

# create and remove
mkdir -p sandbox/logs
touch sandbox/app.log
cp sandbox/app.log sandbox/app.log.bak
mv sandbox/app.log sandbox/app-2025-10-01.log
rm -i sandbox/app.log.bak   # -i prompts before delete
rm -r sandbox/logs          # remove directory tree

# print text into a file
echo "hello world" > notes.txt
cat notes.txt
```

Day 2: Unix Commands — Basics II

Goals: Search files and contents, view parts of files, and manage permissions/ownership.

Mandatory commands & concepts:

- grep, find, wc, file, stat, head, tail, less, chmod, chown, umask, df, du

- Permission bits: rwx for user/group/others; numeric modes 644/755; recursive apply with -R.
- Process quick checks: ps, top (optional).

Examples:

```
# search text recursively (case-insensitive) and show line numbers
grep -Rin "ERROR" /var/log

# find python files > 1MB and list
find . -name "*.py" -size +1M -print

# check file type & counts
file /bin/bash
wc -l access.log

# first/last lines; follow log
head -n 20 access.log
tail -n 50 access.log
tail -f access.log

# permissions (u=user, g=group, o=others)
chmod 644 notes.txt        # rw-r--r--
chmod -R 755 scripts/      # rwxr-xr-x for dirs/files
sudo chown user:group notes.txt

# disk usage
df -h
du -sh .  # total size of current tree
```

Day 3: Vi/Vim Editor — Essentials

Goals: Open, edit, search/replace, and save files confidently.

- Cheat-sheet:
- Modes: ESC (Normal), i/a/o (Insert), v/V/Ctrl+v (Visual).
- Movement: h j k l; w/b words; gg/G file start/end; :set number; /pattern to search.
- Editing: dd (delete line), yy (yank), p (paste), u (undo), Ctrl+r (redo).
- Save/quit: :w, :q, :wq, :q! (force quit).
- Search & Replace:

```
:%s/old/new/g          " replace all
:%s/old/new/gc         " replace with confirm
:1,20s/foo/bar/g       " only in lines 1..20
```

Day 4: SQL — DDL (Schemas & Constraints)

Goals: Design tables with proper data types and constraints.

- Mandatory topics:
- CREATE/ALTER/DROP/TRUNCATE/RENAME

- Constraints: NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, CHECK, DEFAULT, INDEX

Examples:

```
-- create
CREATE TABLE customers (
  customer_id    NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
  first_name     VARCHAR2(50) NOT NULL,
  last_name      VARCHAR2(50) NOT NULL,
  email          VARCHAR2(120) UNIQUE,
  status         VARCHAR2(10) DEFAULT 'ACTIVE' CHECK (status IN
('ACTIVE','INACTIVE')),
  created_at     TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- alter
ALTER TABLE customers ADD (phone VARCHAR2(20));
ALTER TABLE customers MODIFY (email VARCHAR2(150));
ALTER TABLE customers DROP COLUMN phone;

-- rename
RENAME customers TO customers_v1;

-- truncate (fast delete all data, cannot rollback)
TRUNCATE TABLE customers_v1;

-- drop
DROP TABLE customers_v1;
```

Day 5: SQL — DML/DRL/DCL/TCL

Goals: Insert/update/delete rows safely, read with SELECT, control transactions & privileges.

Examples:

```
-- DML
INSERT INTO customers (first_name, last_name, email)
VALUES ('Ana','Sharma','ana@example.com');

UPDATE customers SET status='INACTIVE' WHERE email='ana@example.com';
DELETE FROM customers WHERE customer_id=10;

-- MERGE (upsert)
MERGE INTO customers c
USING (SELECT 'raj@example.com' email FROM dual) s
ON (c.email = s.email)
WHEN NOT MATCHED THEN
  INSERT (first_name, last_name, email) VALUES ('Raj','K','raj@example.com');

-- DRL (SELECT)
SELECT customer_id, first_name, email
FROM customers
WHERE status='ACTIVE'
ORDER BY created_at DESC
FETCH FIRST 10 ROWS ONLY;
```

```
-- TCL
SAVEPOINT before_bulk;
DELETE FROM customers WHERE status='INACTIVE';
ROLLBACK TO before_bulk;  -- undo the delete

COMMIT;  -- persist changes

-- DCL
GRANT SELECT ON customers TO report_user;
REVOKE SELECT ON customers FROM report_user;
```

Day 6: SQL — Functions, Aggregations & Joins

- Mandatory functions: NVL/COALESCE, UPPER/LOWER, TRIM, SUBSTR, INSTR, TO_DATE/TO_CHAR, DATEADD (db specific).
- Aggregations: COUNT, SUM, AVG, MIN, MAX with GROUP BY, HAVING.
- Joins: INNER, LEFT/RIGHT OUTER, FULL OUTER, CROSS; ON vs USING.

```
-- string/date helpers
SELECT UPPER(TRIM(first_name)) AS fname_upper,
       SUBSTR(email, 1, INSTR(email,'@')-1) AS email_user
FROM customers;

-- aggregation
SELECT status, COUNT(*) AS cnt
FROM customers
GROUP BY status
HAVING COUNT(*) > 1;

-- joins
SELECT o.order_id, c.first_name, o.total_amount
FROM orders o
JOIN customers c ON c.customer_id = o.customer_id
WHERE o.created_at >= DATE '2025-09-01';
```

Day 7: PL/SQL — Procedures, Functions, Cursors & Exceptions

Goals: Write stored logic with variables, loops, exception handling, and cursors.

Examples:

```
-- procedure
CREATE OR REPLACE PROCEDURE activate_customer(p_email IN VARCHAR2) AS
BEGIN
  UPDATE customers SET status='ACTIVE' WHERE email = p_email;
  COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    ROLLBACK;
    RAISE;
END;
/

-- function
```

```
CREATE OR REPLACE FUNCTION get_active_count RETURN NUMBER AS
  v_cnt NUMBER;
BEGIN
  SELECT COUNT(*) INTO v_cnt FROM customers WHERE status='ACTIVE';
  RETURN v_cnt;
END;
/

-- cursor example
DECLARE
  CURSOR cur IS SELECT customer_id FROM customers WHERE status='ACTIVE';
  v_id customers.customer_id%TYPE;
BEGIN
  OPEN cur;
  LOOP
    FETCH cur INTO v_id;
    EXIT WHEN cur%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE('Active customer: '||v_id);
  END LOOP;
  CLOSE cur;
END;
/

-- exception sample
BEGIN
  INSERT INTO customers(first_name,last_name,email) VALUES (NULL, 'X', 'x@x.x');
EXCEPTION
  WHEN VALUE_ERROR THEN DBMS_OUTPUT.PUT_LINE('Bad value');
  WHEN OTHERS THEN DBMS_OUTPUT.PUT_LINE(SQLERRM);
END;
/
```

Day 8: Finacle Intro - CIF

- Login flows and CIF creation/linkage.
- Key tables: CIF master, account master indexes.
- Checklist: mandatory fields, validations, error codes.

Day 9: Finacle - CASA Account Opening

- Open Savings/Current with validations.
- Track status in relevant tables.
- Data capture best practices.

Day 10: Finacle - TD/OD/DD

- Steps to open & close TD/OD/DD.
- Interest parameters.
- Reversal scenarios.

Day 11: Finacle - Loan Accounts

- Loan creation, schedules.
- Transactions & charges mapping.
- Delinquency basics.

Day 12: Finacle - Scheme Intro

- SOL, GL, GLSH, Place Holder, Scheme mapping.
- Impacts on postings/reporting.

Day 13: Finacle - Transactions

- Cash/transfer/journal entries.
- Reversal & audit trail.
- Posting rules.

Day 14: Finacle - Inventory & Cheque

- Cheque book issuance.
- Leaf management.
- Stop/return flows.

Day 15: Finacle - Charges & Interest

- Periodic charge setups.
- Accrual & booking.
- Verification tables.

Day 16: Finacle - SI & TOD

- Standing instruction lifecycle.
- TOD limits & expiry.

Day 17: Finacle - Multi-Currency

- FX accounts.
- Rates, revaluation & reversals.

Day 18: InterSOL/CCY Transactions

- Inter-branch postings.
- X-currency validations.

Day 19: SOL/Scheme Transfer

- Account migration between SOL/Schemes.
- Audit checks.

Day 20: Teller Account

- Teller postings, overrides & limits.
- End-of-day checks.

Day 21: Clearing

- Inward/Outward flows.
- Return handling & status codes.

Day 22: Payments - Trade Finance

- Instruments overview.
- Basic process lifecycle.

Day 23: Payments - SWIFT/RTGS

- Message types, cutoffs.
- Exception handling.

Day 24: Batch Jobs

- Setup & scheduling.
- Replication monitors.

Day 25: Scripting - Basics

- Script structure, variables, types.
- Loops & branching.
- I/O commands.

Day 26: Scripting - Functions

- MID, STRLEN, GETPOSITION.
- File read/write/append.
- Existence checks.

Day 27: Scripting - Userhooks

- Hook points & use cases.
- Safety & rollback.

Day 28: Scripting - SQL

- Embedded SQL & cursors.
- Dynamic SQL patterns.

Day 29: Scripting - EXEC

- Babx/spbx call patterns.
- Return codes & logs.

Day 30: Workflow Debugging

- finacle.debug, SRV discovery.
- Field tracing.

Day 31: Workflow Values

- Pre/Post values.
- Error propagation.

Day 32: Writing Workflow

- State diagrams → implementation.
- Test cases.

Day 33: ONS Core Customization

- Page discovery.
- Validation & labels.

Day 34: ONS Core Customization

- crit → det data flow.
- Front/back glue.

Day 35: ONS: Frontend↔DB↔Frontend

- Persist & retrieve patterns.
- Idempotency.

Day 36: ONS Custom Menu

- Menu lifecycle & routing.
- Auth & roles.

Day 37: ONS Custom/Core Searcher

- Indexing & filters.
- Perf tips.

Day 38: ONS Menu + JS + Validation

- Client validations + server guards.
- Audit logs.

Day 39: ONS Menu with Service

- Service calls & retries.
- Timeouts.

Day 40: ONS Menu with MultiRec

- Multi-row patterns.
- Concurrency.

Day 41: ONS Menu with MultiTab

- State sync across tabs.
- UX best practices.

Day 42: ONS Menu with MultiPage

- Navigation & session state.
- Back/forward safety.

Day 43: ONS Menu with MRM

- Maker-Checker flows.
- Rework queues.

Day 44: Script Charges/Batch

- Scheduled charges.
- Backfill scripts.

Day 45: Reports: Design

- Layouts, parameters.
- Join strategy.

Day 46: Reports: Deploy

- Promotion steps.
- Config toggles.

Day 47: Reports: Validate/Searcher

- Custom searchers.
- Edge cases.

Day 48: Reports: HPR

- Generate & move to HPR.
- Archival rules.

Day 49: FI Product APIs

- Catalog & auth.
- Throttling.

Day 50: FI Custom APIs

- Extending payloads.
- Versioning.

Day 51: Expose Custom Fields

- API contracts.
- Backfill impacts.

Day 52: FI/CRM Integration

- CIF creation.
- Log paths & trace IDs.

Day 53: Trusted User

- Session creation.
- Least privilege.

Day 54: COM/SH

- COM file syntax.
- I/O passing.

Day 55: Basic Service

- SLAs & escalation.
- Runbooks.

Day 56: CRM Customization (Basic)

- Field rules & validation.
- Additions.

Day 57: CRM Customization (Advanced)

- Mandatory & type changes.
- Labeling.

Day 58: C24 Intro

- Services & logs.
- Debugging.

Day 59: C24 Formats

- Message maps & validation.
- Schema changes.

Day 60: Review & Labs

- Capstone practice.
- Q&A, cheat-sheets.

Appendix A: Unix Quick Reference

```
# compression & archiving
tar -czf logs.tar.gz logs/
tar -xzf logs.tar.gz -C /tmp

# networking
curl -I https://example.com
ping -c 4 8.8.8.8
netstat -tulpen  # or: ss -tulpen
```

```
# environment
export APP_ENV=dev
echo $APP_ENV

# find & exec (delete old logs)
find /var/log -name "*.log" -mtime +30 -exec rm -f {} \;
```

Appendix B: SQL Quick Reference

```
-- window functions
SELECT customer_id, created_at,
       ROW_NUMBER() OVER (PARTITION BY status ORDER BY created_at DESC) rn
FROM customers;

-- conditional aggregation
SELECT SUM(CASE WHEN status='ACTIVE' THEN 1 ELSE 0 END) AS active_cnt,
       SUM(CASE WHEN status='INACTIVE' THEN 1 ELSE 0 END) AS inactive_cnt
FROM customers;
```

Appendix C: PL/SQL Quick Reference

```
-- basic types & control
DECLARE
  v_count NUMBER := 0;
BEGIN
  FOR i IN 1..10 LOOP
    v_count := v_count + i;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE('Sum='||v_count);
END;
/

-- bulk collect
DECLARE
  TYPE t_ids IS TABLE OF NUMBER;
  ids t_ids;
BEGIN
  SELECT customer_id BULK COLLECT INTO ids FROM customers WHERE ROWNUM <= 100;
  FOR i IN 1..ids.COUNT LOOP
    DBMS_OUTPUT.PUT_LINE(ids(i));
  END LOOP;
END;
/
```