Jerin Mulangan
4347.502 KHAN

PROJECT QUESTIONS

1) Five Additional Business Rules

• A customer's preferred sales rep must be the one with the most total sales to that customer.

• An employee cannot work overlapping shifts in different departments.

• Vendors with a credit rating below 500 require manager approval for purchase orders.

• A part's total used amount across all products must not exceed its current stock level.

• Interview rounds for the same candidate/job must have strictly increasing round numbers.

2) Super-class/Sub-class Modeling
The schema I made uses foreign keys and CHECK constraints to mimic the roles that I made in the EER, the overlapping superclass type of PERSON, and its subclasses EMPLOYEE, CUSTOMER, POTENTIAL EMPLOYEE. By allowing the DDL script to inherit the attributes from the PERSON superclass helped me enforce shared defaults, DRY up common attributes, and clarify constraints in Oracle's object-relational extensions. But in plain relational design, separate tables + FKs are simpler and portable.

3) Why Use an RDBMS (e.g. Oracle)?

- ACID guarantees: Ensures safe concurrent access to critical sales, payroll, and HR data.
- Declarative constraints: Built-in PK/FK/unique/check support enforces business rules at the engine level.
- Scalability & tooling: Mature optimizers, indexing, and monitoring in Oracle handle large volumes of transactions.
- Security & auditing: Fine-grained privileges, roles, and audit trails satisfy corporate compliance requirements.

DESCRIPTION OF PROBLEM:

XYZCOMPANY Enterprise Data-management SystemThe goal of this project is to design and implement a robust relational database for XYZCOMPANY, a mid-sized manufacturing and sales organization that needs to track people (employees, customers, potential hires), job postings, interviews, products, sales, and vendor/part procurement.

Key Requirements

Jerin Mulangan
4347.502 KHAN

People are at the core: every individual is a PERSON with identifiers, demographics, and contact info (including multiple phone numbers). They may simultaneously function as employees, customers, or potential employees, with specialized tables capturing rank/title for employees and preferred reps for customers.

Human resources: the system records hierarchical reporting (one supervisor per employee), multi-department assignments over time, and per-shift schedules. Job postings are managed by department, and both existing employees and external candidates (POTENTIAL_EMPLOYEE) may apply. Interview rounds, graded by employee interviewers, determine final selection based on passing ≥5 rounds with average >70.

Sales & marketing: multiple MARKETING_SITEs are staffed by employees; each sale ties a salesman, a customer, a product, a site, and a timestamp; triggers update customers' preferred reps automatically.

Product management & procurement: products have types, sizes, prices, weights, and styles. Vendors supply parts at fixed per-vendor prices; products consume parts in specified quantities.

Compensation tracking: each employee's payroll history is recorded by transaction, date, and amount.

Normalization & IntegrityAll tables are designed in Third Normal Form to eliminate redundancy, enforce full-key dependencies, and prevent update anomalies. Foreign keys, unique constraints, and triggers capture business rules such as age limits (no employee >65), valid site assignments, and integrity of sales and interview workflows.

Technologies & ToolsWe chose a relational DBMS (Oracle/MySQL) for its ACID compliance, declarative constraint support, powerful query optimizer, and enterprise-grade security features. Java/Swing provides a cross-platform console UI with JDBC for executing both ad-hoc SQL and pre-defined analytics queries.

This system will streamline hiring workflows, ensure accurate HR/payroll records, optimize product procurement, and deliver timely sales analytics—all while preserving data integrity and adaptability for future business expansions.

PROJECT SUMMARY

In this project, we designed and implemented a comprehensive relational database for XYZCOMPANY to manage its people, organizational structure, hiring processes, sales operations, and supply-chain activities. Starting from detailed requirements, we modeled a rich Entity–Relationship (ER) diagram capturing Persons (employees, customers, potential employees), Departments, Jobs, Applicants, Interviews, Products, Sites, Vendors, Parts, Sales, Salaries, and multi-valued Phone numbers. We then translated that conceptual model into a fully normalized (3NF) logical schema, enforcing referential integrity with foreign

keys, triggers for business rules (e.g. age checks, sales-site consistency), and views to precompute key metrics (average salary, top-selling products, selected interviewees). Finally, we built a Java Swing console application that allows end users to explore the schema, run both ad-hoc and predefined project queries, and visualize results in a tabular UI.

Key Challenges

Complex relationships: Modeling many-to-many and recursive relationships (e.g. employees supervising employees, staff rotating through multiple departments).

Business-rule enforcement: Implementing triggers to prevent employees over age 65, ensuring sales only at valid site–employee pairs, and maintaining preferred sales-rep logic.

Normalization: Decomposing rich entities (multi-valued phone numbers, polymorphic applicants) into separate tables without losing semantic clarity.

Usability: Integrating JDBC and Swing to provide an intuitive dropdown-based console, while packaging the MySQL driver on the classpath.

Assumptions

A person's age at hire must be under 65; we enforce this both for employees and potential employees.

An employee may work in different departments over time but only one at a time, captured via start/end dates.

Interview rounds are recorded per candidate–job pair; a candidate is "selected" only after passing ≥5 rounds with average grade >70.

Product-part pricing is static per vendor, and each sale involves exactly one product, one customer, one salesman, and one site.

What I Learned

How to translate real-world business rules into robust SQL constraints and triggers.

The discipline of normalization—particularly handling polymorphic relationships (Employee vs. Potential_Employee) and multivalued attributes (Phone_No).

Best practices for designing and indexing a medium-sized transactional schema in MySQL.

Integrating Java's JDBC with Swing to build a lightweight but functional query console that enhances user interaction with the database.

Jerin Mulangan
4347.502 KHAN

FUNCTIONAL DEPENDENCIES

- ➔ PERSON(PERSONAL_ID → FIRST_NAME, LAST_NAME, BIRTH_DATE, GENDER, ADDRESS_LINE_1, ADDRESS_LINE_2, EMAIL, CITY, STATE, ZIP_CODE)
- ➔ DEPARTMENT(DEPARTMENT_ID → DEPARTMENT_NAME)
- ➔ EMPLOYEE(EMPLOYEE_ID → EMPLOYEE_RANK, EMPLOYEE_TITLE, DEPARTMENT_ID, EMPLOYEE_SUPERVISOR)
- ➔ CUSTOMER(CUSTOMER_ID → PREFERRED_SALES_REP)
- ➔ POTENTIAL_EMPLOYEE(POTENTIAL_EMPLOYEE_ID → —)
- ➔ SHIFT((EMPLOYEE_ID,DEPARTMENT_ID,SHIFT_START_TIME) → SHIFT_END_TIME)
- ➔ EMPLOYEE_WORKS_AT_DEPARTMENT((EMPLOYEE_ID,DEPARTMENT_ID,START_DATE) → END_DATE)
- ➔ JOB(JOB_ID → JOB_DESCRIPTION, POSTED_DATE, DEPARTMENT_POST)
- ➔ APPLICANT(APPLICANT_ID → APPLICANT_CATEGORY, EMPLOYEE_ID, POTENTIAL_EMPLOYEE_ID)
- ➔ APPLIED((JOB_ID,APPLICANT_ID) → —)
- ➔ SELECTED_FOR_INTERVIEW(SELECTED_APPLICANT → JOB_ID, APPLICANT_ID)
- ➔ INTERVIEWER(INTERVIEWER_ID → EMPLOYEE_ID)
- ➔ INTERVIEW(INTERVIEW_ID → CANDIDATE_ID, INTERVIEWER_ID, JOB_POSITION, INTERVIEW_TIME, INTERVIEW_ROUND, INTERVIEW_GRADE)
- ➔ PRODUCT(PRODUCT_ID → PRODUCT_TYPE, PRODUCT_SIZE, PRODUCT_LIST_PRICE, PRODUCT_WEIGHT, PRODUCT_STYLE)
- ➔ MARKETING_SITE(SITE_ID → SITE_NAME, SITE_LOCATION)
- ➔ EMPLOYEE_WORKS_FOR_SITE((EMPLOYEE_ID,SITE_ID) → —)
- ➔ PRODUCT_SOLD_AT_SITE((PRODUCT_ID,SITE_ID) → —)
- ➔ SALE(SALE_ID → SALESMAN_ID, CUSTOMER_ID, PRODUCT_ID, SITE_ID, SALES_TIME)

- ➔ CUSTOMER_BUYS_SALE((SALE_ID,CUSTOMER_ID) → —)

- ➔ VENDOR(VENDOR_ID → VENDOR_NAME, VENDOR_ADDRESS_LINE_1, VENDOR_ADDRESS_LINE_2,
  VENDOR_CITY, VENDOR_STATE, VENDOR_ZIP_CODE,
  VENDOR_ACCOUNT_NO, VENDOR_CREDIT_RATING,
  VENDOR_WEBSERVICE_URL)

- ➔ PART(PART_ID → PART_TYPE)

- ➔ PART_SUPPLIED_BY_VENDOR((PART_ID,VENDOR_ID) → PART_PRICE)

- ➔ PART_USED_IN_PRODUCT((PART_ID,PRODUCT_ID) → USED_AMOUNT)

- ➔ SALARY((EMPLOYEE_ID,TRANSACTION_NO) → PAY_DATE, PAY_AMOUNT)

- ➔ PERSON_PHONE_NO((PERSONAL_ID,PHONE_NO) → —)

General criteria for 3NF: A table is in Third Normal Form if

1. It is in Second Normal Form (all non-key attributes fully depend on the primary key).
2. It has no transitive dependencies (non-key attributes depending on other non-key attributes).

Below is a brief explanation for each relation:

1. PERSON (PERSONAL_ID → FIRST_NAME, LAST_NAME, BIRTH_DATE, GENDER, ADDRESS_LINE_1, ADDRESS_LINE_2, EMAIL, CITY, STATE, ZIP_CODE)
   - *PK*: PERSONAL_ID
   - All other columns describe that person; no subset of PERSONAL_ID determines any other column.
   - No non-key attribute → another non-key attribute dependency.
2. DEPARTMENT (DEPARTMENT_ID → DEPARTMENT_NAME)
   - *PK*: DEPARTMENT_ID
   - Single non-key attribute fully depends on ID; no transitive dependencies.
3. EMPLOYEE (EMPLOYEE_ID → EMPLOYEE_RANK, EMPLOYEE_TITLE, DEPARTMENT_ID, EMPLOYEE_SUPERVISOR)
   - *PK*: EMPLOYEE_ID
   - All columns describe that employee.
   - References PERSON via foreign key, but no non-key attribute depends on another non-key attribute.
4. CUSTOMER (CUSTOMER_ID → PREFERRED_SALES_REP)
   - *PK*: CUSTOMER_ID
   - PREFERRED_SALES_REP is a foreign key, no additional attributes exist.
5. POTENTIAL_EMPLOYEE (POTENTIAL_EMPLOYEE_ID)

    - *PK*: POTENTIAL_EMPLOYEE_ID, no other attributes.
6. SHIFT (EMPLOYEE_ID, DEPARTMENT_ID, SHIFT_START_TIME → SHIFT_END_TIME)
    - Composite *PK*: (EMPLOYEE_ID, DEPARTMENT_ID, SHIFT_START_TIME)
    - SHIFT_END_TIME depends only on the full key; no partial/transitive dependencies.
7. EMPLOYEE_WORKS_AT_DEPARTMENT (EMPLOYEE_ID, DEPARTMENT_ID, START_DATE → END_DATE)
    - Composite *PK*: (EMPLOYEE_ID, DEPARTMENT_ID, START_DATE)
    - END_DATE depends on full key; no other non-key columns.
8. JOB (JOB_ID → JOB_DESCRIPTION, POSTED_DATE, DEPARTMENT_POST)
    - *PK*: JOB_ID
    - All non-key attributes describe the job; DEPARTMENT_POST is a FK, no extra dependencies.
9. APPLICANT (APPLICANT_ID → APPLICANT_CATEGORY, EMPLOYEE_ID, POTENTIAL_EMPLOYEE_ID)
    - *PK*: APPLICANT_ID
    - Category and one of the two FKs depend only on APPLICANT_ID.
    - CHECK constraint enforces exactly one of EMPLOYEE_ID or POTENTIAL_EMPLOYEE_ID, but no transitive FDs.
10. APPLIED (JOB_ID, APPLICANT_ID)
    - Composite *PK*: (JOB_ID, APPLICANT_ID), no other columns.
11. SELECTED_FOR_INTERVIEW (SELECTED_APPLICANT → JOB_ID, APPLICANT_ID)
    - *PK*: SELECTED_APPLICANT
    - JOB_ID, APPLICANT_ID depend solely on SELECTED_APPLICANT.
12. INTERVIEWER (INTERVIEWER_ID → EMPLOYEE_ID)
    - *PK*: INTERVIEWER_ID, no transitive dependencies.
13. INTERVIEW (INTERVIEW_ID → CANDIDATE_ID, INTERVIEWER_ID, JOB_POSITION, INTERVIEW_TIME, INTERVIEW_ROUND, INTERVIEW_GRADE)
    - *PK*: INTERVIEW_ID
    - All columns describe that interview event; no non-key dependencies among themselves.
14. PRODUCT (PRODUCT_ID → PRODUCT_TYPE, PRODUCT_SIZE, PRODUCT_LIST_PRICE, PRODUCT_WEIGHT, PRODUCT_STYLE)
    - *PK*: PRODUCT_ID
    - All descriptive attributes fully depend on the key.
15. MARKETING_SITE (SITE_ID → SITE_NAME, SITE_LOCATION)
    - *PK*: SITE_ID, no transitive dependencies.
16. EMPLOYEE_WORKS_FOR_SITE (EMPLOYEE_ID, SITE_ID) no additional attributes, 3NF.
17. PRODUCT_SOLD_AT_SITE (PRODUCT_ID, SITE_ID) no additional attributes, 3NF.
18. SALE (SALE_ID → SALESMAN_ID, CUSTOMER_ID, PRODUCT_ID, SITE_ID, SALES_TIME)
    - *PK*: SALE_ID
    - Foreign keys capture relationships; no non-key attributes depend on each other.
19. CUSTOMER_BUYS_SALE (SALE_ID, CUSTOMER_ID) no additional attributes, 3NF.

20. VENDOR (VENDOR_ID → VENDOR_NAME, ADDRESS_LINE_1, ADDRESS_LINE_2, CITY, STATE, ZIP_CODE, ACCOUNT_NO, CREDIT_RATING, WEBSERVICE_URL)
    - *PK*: VENDOR_ID; descriptive attributes fully depend on it.
21. PART (PART_ID → PART_TYPE)
    - *PK*: PART_ID, no extra dependencies.
22. PART_SUPPLIED_BY_VENDOR (PART_ID, VENDOR_ID → PART_PRICE)
    - Composite *PK*: (PART_ID, VENDOR_ID)
    - PART_PRICE depends on both; no transitive FDs.
23. PART_USED_IN_PRODUCT (PART_ID, PRODUCT_ID → USED_AMOUNT)
    - Composite *PK*: (PART_ID, PRODUCT_ID)
    - USED_AMOUNT depends on both; no transitive FDs.
24. SALARY (EMPLOYEE_ID, TRANSACTION_NO → PAY_DATE, PAY_AMOUNT)
    - Composite *PK*: (EMPLOYEE_ID, TRANSACTION_NO)
    - PAY_DATE and PAY_AMOUNT depend on both; transaction_no is only unique per employee.
25. PERSON_PHONE_NO (PERSONAL_ID, PHONE_NO) no other attributes, 3NF.

Jerin Mulangan
4347.502 KHAN

ER MODEL

Entities

Person with attributes Personal_ID (PK), First_Name, Last_Name, Birth_Date, Gender, Address_Line_1/2, City, State, Zip_Code.

Department, Job, Product, Marketing_Site, Vendor, Part, Sale, Salary, Phone (multivalued).

Relationships

Employee (a Person) works_in Department (many-to-many over time, tracked by Employee_Works_At_Department).

Employee supervises Employee (recursive).

Customer (a Person) buys Sale; has a preferred sales rep (an Employee).

Sale ties Employee, Customer, Product, and Marketing_Site together.

Vendor supplies Part; Part used_in Product.

Applicant (an Employee or Potential_Employee) applies_to Job; selected applicants interviewed_by Interviewer (Employee).

Interview records rounds & grades.

LOGICAL MODEL

Table Key FKs

PERSON PERSONAL_ID —

EMPLOYEE EMPLOYEE_ID → PERSON, → DEPARTMENT, → EMPLOYEE (supervisor)

CUSTOMER CUSTOMER_ID → PERSON, → EMPLOYEE (preferred sales rep)

POTENTIAL_EMPLOYEE POTENTIAL_EMPLOYEE_ID → PERSON

APPLICANT APPLICANT_ID → EMPLOYEE or → POTENTIAL_EMPLOYEE

APPLIED (JOB_ID, APPLICANT_ID) → JOB, → APPLICANT

SELECTED_FOR_INTERVIEWSELECTED_APPLICANT → APPLIED

INTERVIEWERINTERVIEWER_ID → EMPLOYEE

INTERVIEW INTERVIEW_ID → SELECTED_FOR_INTERVIEW, → INTERVIEWER, → JOB

PRODUCT, PART, VENDOR PRODUCT_ID, PART_ID, VENDOR_ID — / → VENDOR / —

PART_SUPPLIED_BY_VENDOR (PART_ID, VENDOR_ID) → PART, → VENDOR

PART_USED_IN_PRODUCT (PART_ID, PRODUCT_ID) → PART, → PRODUCT

MARKETING_SITE, SALE SITE_ID / SALE_ID → EMPLOYEE_WORKS_FOR_SITE, → PRODUCT_SOLD_AT_SITE, → CUSTOMER

EMPLOYEE_WORKS_FOR_SITE (EMPLOYEE_ID, SITE_ID) → EMPLOYEE, → MARKETING_SITE

EMPLOYEE_WORKS_AT_DEPARTMENT (EMPLOYEE_ID, DEPARTMENT_ID, START_DATE) → EMPLOYEE, → DEPARTMENT

SHIFT (EMPLOYEE_ID, DEPARTMENT_ID, SHIFT_START_TIME) → EMPLOYEE, → DEPARTMENT

SALARY (EMPLOYEE_ID, TRANSACTION_NO) → EMPLOYEE

PERSON_PHONE_NO (PERSONAL_ID, PHONE_NO) → PERSON

EER MODEL

The EER model enriches the ER with:

Specialization (ISA) hierarchies

Person is a super-class; Employee, Customer, and Potential_Employee are sub-classes → captures rule that a person can belong to multiple categories simultaneously.

Multivalued attribute

Phone_No as a separate table to handle multiple phone numbers per person.

Weak entity

Applied, Selected_for_Interview, and Interview depend on parent entities; their PKs include FKs.

Ternary/Associative relationships

Sale associates Employee, Customer, Product, and Site; implemented in SALE table with composite FKs.