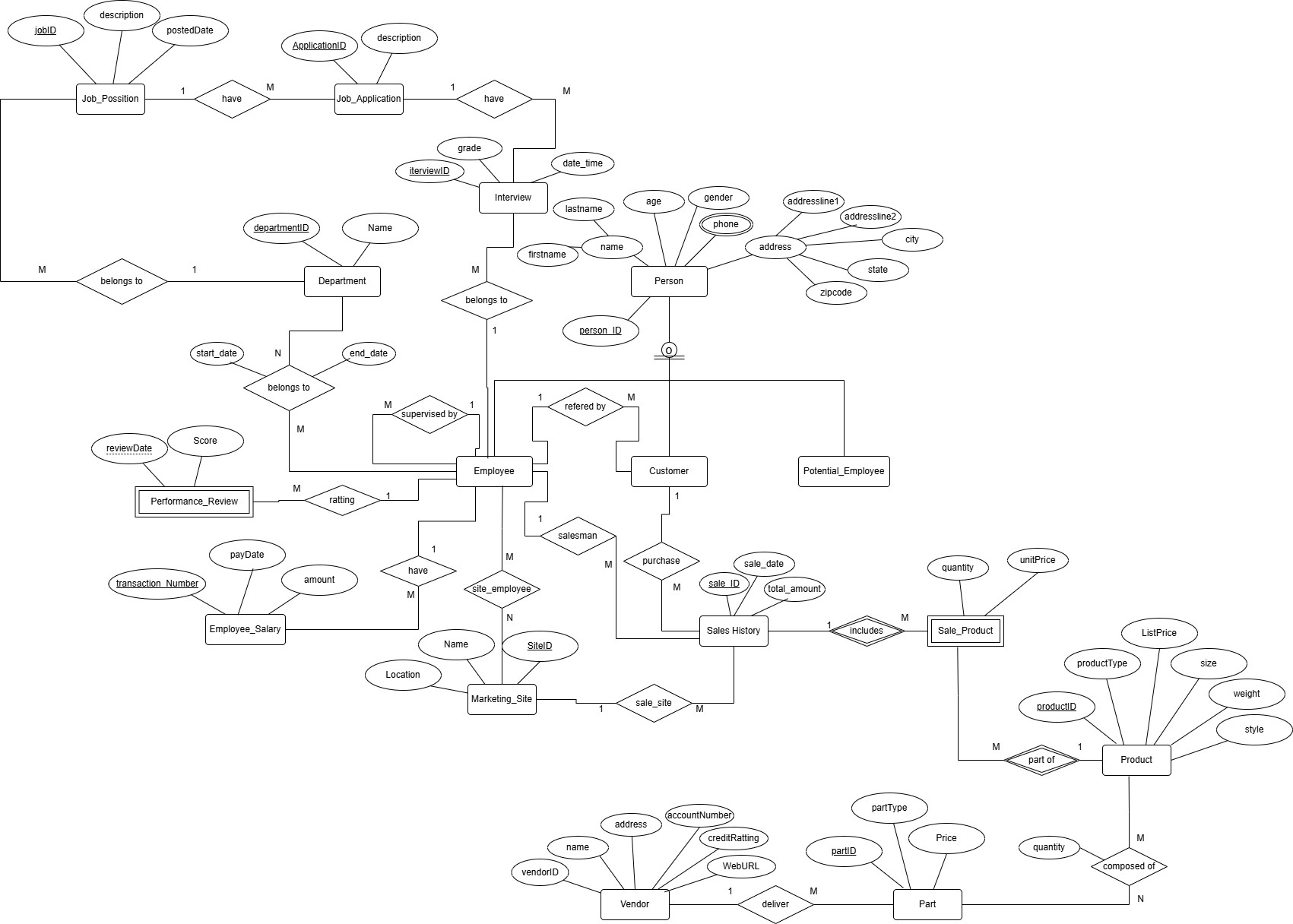
CS 4347.502- Semester Project - Part 2( Final)

Arafat Khan, Axk22019

1. **EER- Diagram**

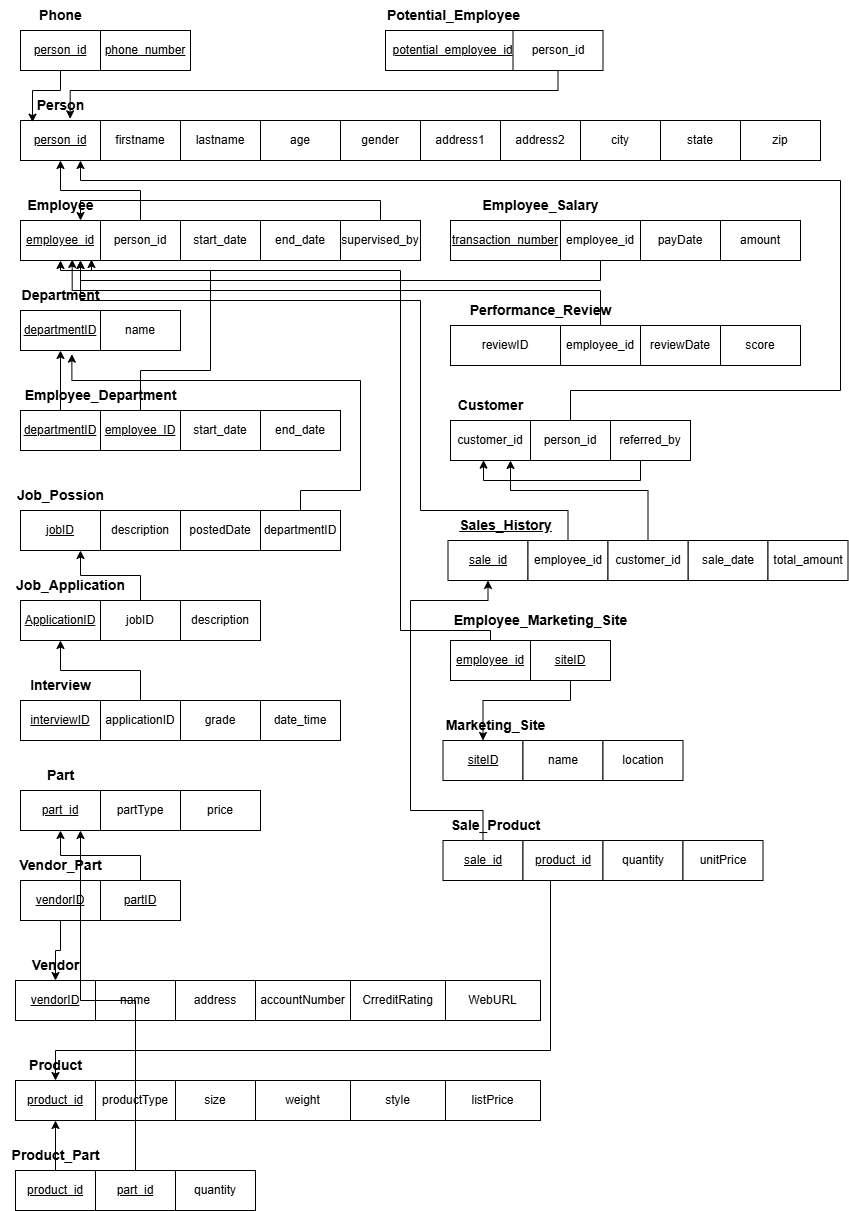


1. **Logical Design (with data types)**

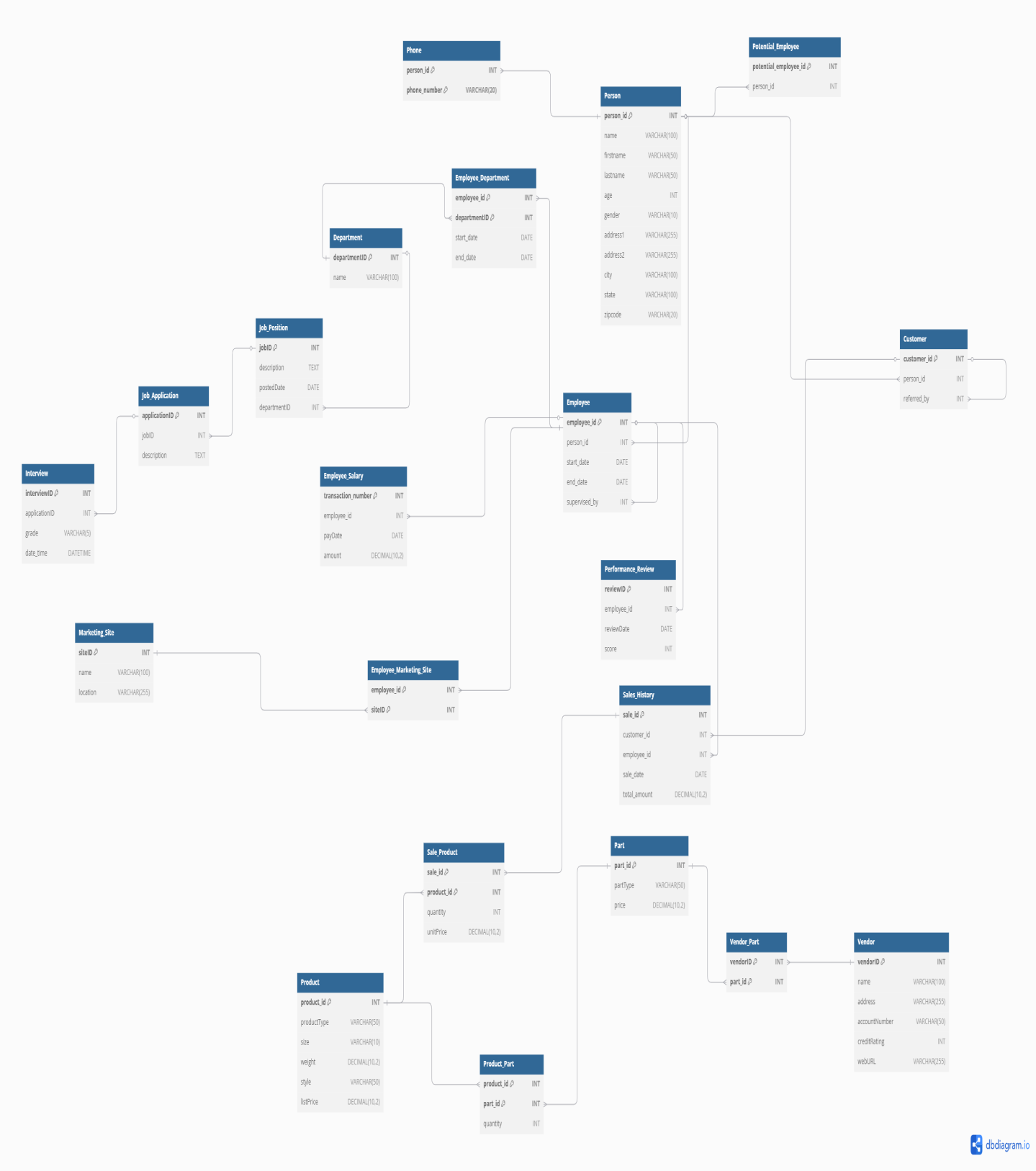
**Assumptions:**

- I did not make any new table for address table.

- I used some my own created primary keys for weak entities. Like in Employee\_performance table and other weak entities.



1. **Physical Design (data dictionary form)**



## Person Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| person\_id | INT | - | Unique identifier for a person | PRIMARY KEY |
| name | VARCHAR | 100 | Full name of the person | - |
| firstname | VARCHAR | 50 | First name of the person | - |
| lastname | VARCHAR | 50 | Last name of the person | - |
| age | INT | - | Age of the person | - |
| gender | VARCHAR | 10 | Gender of the person | - |
| address1 | VARCHAR | 255 | Primary address line | - |
| address2 | VARCHAR | 255 | Secondary address line | - |
| city | VARCHAR | 100 | City of residence | - |
| state | VARCHAR | 100 | State of residence | - |
| zipcode | VARCHAR | 20 | Postal/ZIP code | - |

## Phone Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| person\_id | INT | - | Reference to person | PRIMARY KEY, FOREIGN KEY references Person(person\_id) |
| phone\_number | VARCHAR | 20 | Phone number | PRIMARY KEY |

## Employee Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| employee\_id | INT | - | Unique employee identifier | PRIMARY KEY |
| person\_id | INT | - | Reference to person | FOREIGN KEY references Person(person\_id) |
| start\_date | DATE | - | Employment start date | - |
| end\_date | DATE | - | Employment end date | - |
| supervised\_by | INT | - | Employee's supervisor | FOREIGN KEY references Employee(employee\_id) |

## Customer Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| customer\_id | INT | - | Unique customer identifier | PRIMARY KEY |
| person\_id | INT | - | Reference to person | FOREIGN KEY references Person(person\_id) |
| referred\_by | INT | - | Customer who referred this customer | FOREIGN KEY references Customer(customer\_id) |

## Potential\_Employee Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| potential\_employee\_id | INT | - | Unique identifier | PRIMARY KEY |
| person\_id | INT | - | Reference to person | FOREIGN KEY references Person(person\_id) |

## Job\_Position Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| jobID | INT | - | Unique job identifier | PRIMARY KEY |
| description | TEXT | - | Job description | - |
| postedDate | DATE | - | Date job was posted | - |
| departmentiD | INT | - | Department ID | FOREIGN KEY references Department(departmentiD) |

## Job\_Application Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| applicationID | INT | - | Unique application identifier | PRIMARY KEY |
| jobID | INT | - | Reference to job position | FOREIGN KEY references Job\_Position(jobID) |
| description | TEXT | - | Application details | - |

## Interview Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| interviewID | INT | - | Unique interview identifier | PRIMARY KEY |
| applicationID | INT | - | Reference to job application | FOREIGN KEY references Job\_Application(applicationID) |
| grade | VARCHAR | 5 | Interview grade/rating | - |
| date\_time | DATETIME | - | Interview date and time | - |

## Department Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| departmentID | INT | - | Unique department identifier | PRIMARY KEY |
| name | VARCHAR | 100 | Department name | - |

## Employee\_Department Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| employee\_id | INT | - | Reference to employee | PRIMARY KEY, FOREIGN KEY references Employee(employee\_id) |
| departmentID | INT | - | Reference to department | PRIMARY KEY, FOREIGN KEY references Department(departmentID) |
| start\_date | DATE | - | Assignment start date | - |
| end\_date | DATE | - | Assignment end date | - |

## Performance\_Review Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| reviewID | INT | - | Unique review identifier | PRIMARY KEY |
| employee\_id | INT | - | Reference to employee | FOREIGN KEY references Employee(employee\_id) |
| reviewDate | DATE | - | Review date | - |
| score | INT | - | Performance score | - |

## Employee\_Salary Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| transaction\_number | INT | - | Unique transaction identifier | PRIMARY KEY |
| employee\_id | INT | - | Reference to employee | FOREIGN KEY references Employee(employee\_id) |
| payDate | DATE | - | Payment date | - |
| amount | DECIMAL | 10,2 | Payment amount | - |

## Marketing\_Site Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| siteID | INT | - | Unique site identifier | PRIMARY KEY |
| name | VARCHAR | 100 | Site name | - |
| location | VARCHAR | 255 | Site location | - |

## Employee\_Marketing\_Site Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| employee\_id | INT | - | Reference to employee | PRIMARY KEY, FOREIGN KEY references Employee(employee\_id) |
| siteID | INT | - | Reference to marketing site | PRIMARY KEY, FOREIGN KEY references Marketing\_Site(siteID) |

## Sales\_History Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| sale\_id | INT | - | Unique sale identifier | PRIMARY KEY |
| customer\_id | INT | - | Reference to customer | FOREIGN KEY references Customer(customer\_id) |
| employee\_id | INT | - | Reference to employee | FOREIGN KEY references Employee(employee\_id) |
| sale\_date | DATE | - | Sale date | - |
| total\_amount | DECIMAL | 10,2 | Total sale amount | - |

## Sale\_Product Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| sale\_id | INT | - | Reference to sale | PRIMARY KEY, FOREIGN KEY references Sales\_History(sale\_id) |
| product\_id | INT | - | Reference to product | PRIMARY KEY, FOREIGN KEY references Product(product\_id) |
| quantity | INT | - | Quantity sold | - |
| unitPrice | DECIMAL | 10,2 | Unit price at time of sale | - |

## Product Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| product\_id | INT | - | Unique product identifier | PRIMARY KEY |
| productType | VARCHAR | 50 | Type of product | - |
| size | VARCHAR | 10 | Product size | - |
| weight | DECIMAL | 10,2 | Product weight | - |
| style | VARCHAR | 50 | Product style | - |
| listPrice | DECIMAL | 10,2 | Standard list price | - |

## Product\_Part Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| product\_id | INT | - | Reference to product | PRIMARY KEY, FOREIGN KEY references Product(product\_id) |
| part\_id | INT | - | Reference to part | PRIMARY KEY, FOREIGN KEY references Part(part\_id) |
| quantity | INT | - | Quantity of part in product | - |

## Part Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| part\_id | INT | - | Unique part identifier | PRIMARY KEY |
| partType | VARCHAR | 50 | Type of part | - |
| price | DECIMAL | 10,2 | Part price | - |

## Vendor Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| vendorID | INT | - | Unique vendor identifier | PRIMARY KEY |
| name | VARCHAR | 100 | Vendor name | - |
| address | VARCHAR | 255 | Vendor address | - |
| accountNumber | VARCHAR | 50 | Account number | - |
| creditRating | INT | - | Credit  rating | - |
| webURL | VARCHAR | 255 | Website URL | - |

## Vendor\_Part Table

| **Attribute** | **Data Type** | **Size** | **Description** | **Constraints** |
| --- | --- | --- | --- | --- |
| vendorID | INT | - | Reference to vendor | PRIMARY KEY, FOREIGN KEY references Vendor(vendorID) |
| part\_id | INT | - | Reference to part | PRIMARY KEY, FOREIGN KEY references Part(part\_id) |

**Phase 03:**

**a) Normalization to 3NF**

You have already implemented most tables in 3NF as shown in your Phase 2. Below is a confirmation and rationale for why your tables are in 3NF

Example:

Employee Table

* 1NF: Atomic attributes only (e.g., start\_date, end\_date are atomic).
* 2NF: Every non-key attribute is fully functionally dependent on the primary key employee\_id.
* 3NF: No transitive dependency exists (e.g., supervised\_by is a foreign key, not a derived or transitive field).

Similar reasoning holds for:

* Person, Phone, Customer, Potential\_Employee
* Department, Employee\_Department, Job\_Position, Job\_Application, Interview
* Product, Sale\_Product, Sales\_History, Part, Vendor, Vendor\_Part, etc.

### ****b) Dependency Diagrams****

Here are **sample dependency diagrams** for a few major tables. Each one shows **primary key → all other non-key attributes**.

**Employee Table**

employee\_id → person\_id, start\_date, end\_date, supervised\_by

**Product Table**

product\_id → productType, size, weight, style, listPrice

**Employee\_Department Table**

(employee\_id, departmentID) → start\_date, end\_date

**Interview Table**

interviewID → applicationID, grade, date\_time

**Vendor\_Part Table**

(vendorID, part\_id) → (No non-key attributes)

**c) SQL Statements**

CREATE DATABASE IF NOT EXISTS phase3;

USE phase3;

-- Person Table (Core entity)

CREATE TABLE Person (

person\_id INT PRIMARY KEY,

name VARCHAR(100),

firstname VARCHAR(50),

lastname VARCHAR(50),

age INT,

gender VARCHAR(10),

address1 VARCHAR(255),

address2 VARCHAR(255),

city VARCHAR(100),

state VARCHAR(100),

zipcode VARCHAR(20),

email VARCHAR(100)

);

-- Phone Numbers

CREATE TABLE Phone (

person\_id INT,

phone\_number VARCHAR(20),

PRIMARY KEY (person\_id, phone\_number),

FOREIGN KEY (person\_id) REFERENCES Person(person\_id)

);

-- Employees and Supervision Structure

CREATE TABLE Employee (

employee\_id INT PRIMARY KEY,

person\_id INT,

start\_date DATE,

end\_date DATE,

supervised\_by INT,

FOREIGN KEY (person\_id) REFERENCES Person(person\_id),

FOREIGN KEY (supervised\_by) REFERENCES Employee(employee\_id)

);

-- Customers and Referrals

CREATE TABLE Customer (

customer\_id INT PRIMARY KEY,

person\_id INT,

referred\_by INT,

FOREIGN KEY (person\_id) REFERENCES Person(person\_id),

FOREIGN KEY (referred\_by) REFERENCES Customer(customer\_id)

);

-- Potential Employees

CREATE TABLE Potential\_Employee (

potential\_employee\_id INT PRIMARY KEY,

person\_id INT,

FOREIGN KEY (person\_id) REFERENCES Person(person\_id)

);

-- Job Positions

CREATE TABLE Job\_Position (

jobID INT PRIMARY KEY,

description TEXT,

postedDate DATE

);

-- Job Applications

CREATE TABLE Job\_Application (

applicationID INT PRIMARY KEY,

jobID INT,

person\_id INT,

description TEXT,

FOREIGN KEY (jobID) REFERENCES Job\_Position(jobID),

FOREIGN KEY (person\_id) REFERENCES Person(person\_id)

);

-- Interviews

CREATE TABLE Interview (

interviewID INT PRIMARY KEY,

applicationID INT,

grade VARCHAR(5),

date\_time DATETIME,

FOREIGN KEY (applicationID) REFERENCES Job\_Application(applicationID)

);

-- Interviewers

CREATE TABLE Interviewer (

interviewer\_id INT PRIMARY KEY,

employee\_id INT,

FOREIGN KEY (employee\_id) REFERENCES Employee(employee\_id)

);

-- Mapping Interviewers to Interviews

CREATE TABLE Interview\_Interviewer (

interview\_id INT,

interviewer\_id INT,

PRIMARY KEY (interview\_id, interviewer\_id),

FOREIGN KEY (interview\_id) REFERENCES Interview(interviewID),

FOREIGN KEY (interviewer\_id) REFERENCES Interviewer(interviewer\_id)

);

-- Departments

CREATE TABLE Department (

departmentID INT PRIMARY KEY,

name VARCHAR(100)

);

-- Employee-Department Assignments

CREATE TABLE Employee\_Department (

employee\_id INT,

departmentID INT,

start\_date DATE,

end\_date DATE,

PRIMARY KEY (employee\_id, departmentID),

FOREIGN KEY (employee\_id) REFERENCES Employee(employee\_id),

FOREIGN KEY (departmentID) REFERENCES Department(departmentID)

);

-- Performance Reviews

CREATE TABLE Performance\_Review (

reviewID INT PRIMARY KEY,

employee\_id INT,

reviewDate DATE,

score INT,

FOREIGN KEY (employee\_id) REFERENCES Employee(employee\_id)

);

-- Salaries

CREATE TABLE Employee\_Salary (

transaction\_number INT PRIMARY KEY,

employee\_id INT,

payDate DATE,

amount DECIMAL(10, 2),

FOREIGN KEY (employee\_id) REFERENCES Employee(employee\_id)

);

-- Marketing Sites

CREATE TABLE Marketing\_Site (

siteID INT PRIMARY KEY,

name VARCHAR(100),

location VARCHAR(255)

);

-- Employee assignments to sites

CREATE TABLE Employee\_Marketing\_Site (

employee\_id INT,

siteID INT,

PRIMARY KEY (employee\_id, siteID),

FOREIGN KEY (employee\_id) REFERENCES Employee(employee\_id),

FOREIGN KEY (siteID) REFERENCES Marketing\_Site(siteID)

);

-- Products

CREATE TABLE Product (

product\_id INT PRIMARY KEY,

productType VARCHAR(50),

size VARCHAR(10),

weight DECIMAL(10, 2),

style VARCHAR(50),

listPrice DECIMAL(10, 2)

);

-- Parts

CREATE TABLE Part (

part\_id INT PRIMARY KEY,

partType VARCHAR(50),

price DECIMAL(10, 2)

);

-- Product-Part Mapping

CREATE TABLE Product\_Part (

product\_id INT,

part\_id INT,

quantity INT,

PRIMARY KEY (product\_id, part\_id),

FOREIGN KEY (product\_id) REFERENCES Product(product\_id),

FOREIGN KEY (part\_id) REFERENCES Part(part\_id)

);

-- Vendors

CREATE TABLE Vendor (

vendorID INT PRIMARY KEY,

name VARCHAR(100),

address VARCHAR(255),

accountNumber VARCHAR(50),

creditRating INT,

webURL VARCHAR(255)

);

-- Vendor-Part Mapping

CREATE TABLE Vendor\_Part (

vendorID INT,

part\_id INT,

PRIMARY KEY (vendorID, part\_id),

FOREIGN KEY (vendorID) REFERENCES Vendor(vendorID),

FOREIGN KEY (part\_id) REFERENCES Part(part\_id)

);

CREATE TABLE Sales\_History (

sale\_id INT PRIMARY KEY,

customer\_id INT,

employee\_id INT,

sale\_date DATE,

total\_amount DECIMAL(10, 2),

FOREIGN KEY (customer\_id) REFERENCES Customer(customer\_id),

FOREIGN KEY (employee\_id) REFERENCES Employee(employee\_id)

);

CREATE TABLE Sale\_Product (

sale\_id INT,

product\_id INT,

quantity INT,

unitPrice DECIMAL(10, 2),

PRIMARY KEY (sale\_id, product\_id),

FOREIGN KEY (sale\_id) REFERENCES Sales\_History(sale\_id),

FOREIGN KEY (product\_id) REFERENCES Product(product\_id)

);

INSERT INTO Person VALUES

(1, 'John Smith', 'John', 'Smith', 35, 'Male', '123 Main St', '', 'New York', 'NY', '10001', 'john.smith@example.com'),

(2, 'Alice Johnson', 'Alice', 'Johnson', 29, 'Female', '456 Oak Ave', '', 'Chicago', 'IL', '60616', 'alice.johnson@example.com'),

(3, 'Bob Williams', 'Bob', 'Williams', 40, 'Male', '789 Pine Rd', '', 'Dallas', 'TX', '75201', 'bob.williams@example.com'),

(4, 'Carol Davis', 'Carol', 'Davis', 33, 'Female', '321 Elm St', '', 'Miami', 'FL', '33101', 'carol.davis@example.com'),

(5, 'Ethan Brown', 'Ethan', 'Brown', 28, 'Male', '654 Cedar Blvd', '', 'Seattle', 'WA', '98101', 'ethan.brown@example.com'),

(6, 'Hellen Cole', 'Hellen', 'Cole', 30, 'Female', '789 Sunset Blvd', '', 'Los Angeles', 'CA', '90001', 'hellen.cole@example.com');

INSERT INTO Phone VALUES

(1, '555-1111'),

(2, '555-2222'),

(3, '555-3333'),

(4, '555-4444'),

(5, '555-5555');

INSERT INTO Employee VALUES

(101, 1, '2020-01-10', NULL, NULL),

(102, 2, '2021-03-15', NULL, 101),

(103, 3, '2019-06-20', NULL, 101),

(104, 4, '2022-09-05', NULL, 102),

(105, 5, '2023-07-12', NULL, 102),

(106, 6, '2023-01-01', NULL, NULL);

INSERT INTO Employee\_Salary VALUES

(1001, 101, '2024-01-30', 5000.00),

(1002, 102, '2024-01-30', 4500.00),

(1003, 103, '2024-01-30', 4800.00),

(1004, 104, '2024-01-30', 4600.00),

(1005, 105, '2024-01-30', 4700.00);

INSERT INTO Customer VALUES

(201, 1, NULL),

(202, 2, 201),

(203, 3, 201),

(204, 4, 202),

(205, 5, 203);

INSERT INTO Potential\_Employee VALUES

(301, 1),

(302, 2),

(303, 3),

(304, 4),

(305, 5);

INSERT INTO Job\_Position VALUES

(401, 'Software Developer', '2024-12-01'),

(402, 'Sales Associate', '2024-12-05'),

(403, 'Marketing Analyst', '2024-12-10'),

(11111, 'Senior Designer', '2025-01-01'),

(403, 'Marketing Analyst', '2024-12-10'),

(11111, 'Senior Designer', '2025-01-01'),

(9999, 'Digital Marketer', '2011-01-15', 703),

(12345, 'Test Job', '2025-05-01', 701);

INSERT INTO Job\_Application VALUES

(999, 12345, 1, 'Testing employee as applicant'),

(501, 401, 1, 'Applied via portal'),

(502, 401, 2, 'Referred by employee'),

(503, 402, 3, 'Walk-in interview'),

(504, 403, 4, 'University job fair'),

(601, 11111, 1, 'Applied for Senior Designer');

INSERT INTO Interview VALUES

(601, 501, 'A+', '2025-01-10 10:00:00'),

(602, 502, 'B', '2025-01-12 14:00:00'),

(603, 503, 'A', '2025-01-15 11:00:00'),

(604, 504, 'C', '2025-01-20 13:00:00'),

(701, 601, 'A', '2025-05-06 10:00:00');

INSERT INTO Department VALUES

(701, 'IT'),

(702, 'Sales'),

(703, 'Marketing');

INSERT INTO Employee\_Department VALUES

(101, 701, '2020-01-10', NULL),

(102, 701, '2021-03-15', NULL),

(103, 702, '2019-06-20', NULL),

(104, 702, '2022-09-05', NULL),

(105, 703, '2023-07-12', NULL),

(101, 702, '2020-01-10', NULL),

(101, 703, '2020-01-10', NULL);

INSERT INTO Performance\_Review VALUES

(801, 101, '2024-12-01', 90),

(802, 102, '2024-12-01', 85),

(803, 103, '2024-12-01', 88),

(804, 104, '2024-12-01', 75),

(805, 105, '2024-12-01', 80);

INSERT INTO Marketing\_Site VALUES

(901, 'Site A', 'New York'),

(902, 'Site B', 'Chicago');

INSERT INTO Employee\_Marketing\_Site VALUES

(105, 901),

(104, 902);

INSERT INTO Product VALUES

(1001, 'Shirt', 'M', 0.5, 'Casual', 29.99),

(1002, 'Pants', 'L', 0.8, 'Formal', 49.99),

(1003, 'Jacket', 'XL', 1.2, 'Winter', 89.99);

INSERT INTO Part VALUES

(1101, 'Button', 0.10),

(1102, 'Zipper', 0.50),

(1103, 'Cloth Panel', 1.00),

(1201, 'Cup', 3.50);

INSERT INTO Product\_Part VALUES

(1001, 1101, 5),

(1002, 1102, 1),

(1003, 1103, 3);

INSERT INTO Vendor VALUES

(1201, 'Best Parts Co.', '123 Supply Rd', 'ACC123', 5, 'http://bestparts.com'),

(1202, 'Zippers Inc.', '456 Fashion Ln', 'ACC456', 4, 'http://zippersinc.com');

INSERT INTO Vendor\_Part VALUES

(1201, 1101),

(1202, 1102),

(1201, 1103),

(1204, 1201);

INSERT INTO Sales\_History VALUES

(1301, 201, 101, '2025-01-25', 129.97),

(1302, 202, 102, '2025-01-26', 89.99);

INSERT INTO Sale\_Product VALUES

(1301, 1001, 1, 29.99),

(1301, 1002, 1, 49.99),

(1301, 1003, 1, 49.99),

(1302, 1003, 1, 89.99);

INSERT INTO Interviewer VALUES

(6, 106);

INSERT INTO Interview\_Interviewer VALUES

(701, 6);

(801, 601, 'A+', '2025-05-01 10:00:00'),

(802, 601, 'A', '2025-05-02 10:00:00'),

(803, 601, 'A', '2025-05-03 10:00:00'),

(804, 601, 'B', '2025-05-04 10:00:00'),

(805, 601, 'A+', '2025-05-05 10:00:00')

**d)View creation statements**

**View 1: Average salary each employee has earned monthly since joining**

CREATE VIEW View1 AS

SELECT

e.employee\_id,

ROUND(AVG(s.amount), 2) AS avg\_monthly\_salary

FROM

Employee\_Salary s

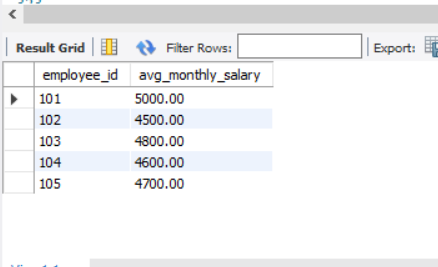
JOIN

Employee e ON e.employee\_id = s.employee\_id

GROUP BY

e.employee\_id;

SELECT \* FROM View1;



**View 2: Number of interview rounds each interviewee passes for each job**

CREATE OR REPLACE VIEW View2 AS

SELECT

ja.applicationID,

COUNT(\*) AS passed\_rounds

FROM

Interview i

JOIN

Job\_Application ja ON ja.applicationID = i.applicationID

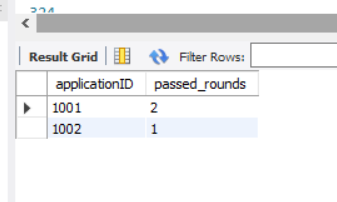
WHERE

i.grade > 60

GROUP BY

ja.applicationID;

SELECT \* FROM View2;



**View 3: Number of items of each product type sold**

CREATE VIEW View3 AS

SELECT

p.productType,

SUM(sp.quantity) AS total\_items\_sold

FROM

Sale\_Product sp

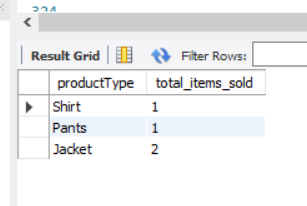
JOIN

Product p ON p.product\_id = sp.product\_id

GROUP BY

p.productType;

SELECT \* FROM View3;

****

**View 4: Part purchase cost for each product**

CREATE VIEW View4 AS

SELECT

pp.product\_id,

SUM(pp.quantity \* pt.price) AS total\_part\_cost

FROM

Product\_Part pp

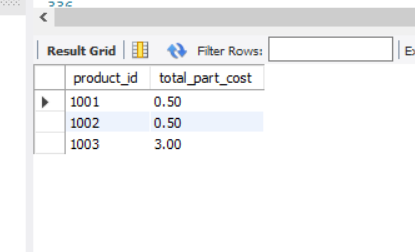
JOIN

Part pt ON pt.part\_id = pp.part\_id

GROUP BY

pp.product\_id;

SELECT \* FROM View4;

****

**SQL Queries**

**Q1: Interviewers for “Hellen Cole” and job “11111”**

SELECT e.employee\_id, p.name

FROM Interview i

JOIN Interview\_Interviewer ii ON i.interviewID = ii.interview\_id

JOIN Interviewer ir ON ii.interviewer\_id = ir.interviewer\_id

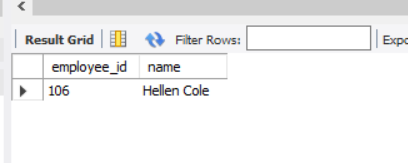
JOIN Employee e ON ir.employee\_id = e.employee\_id

JOIN Person p ON e.person\_id = p.person\_id

JOIN Job\_Application ja ON i.applicationID = ja.applicationID

JOIN Job\_Position jp ON ja.jobID = jp.jobID

WHERE jp.jobID = 11111 AND p.name = 'Hellen Cole';



**Q2: Jobs posted by “Marketing” in Jan 2011**

SELECT jp.jobID

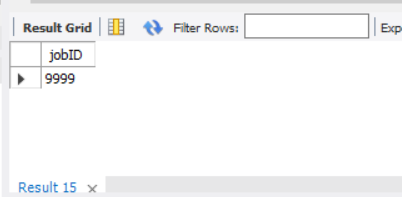
FROM Job\_Position jp

JOIN Department d ON jp.departmentID = d.departmentID

WHERE d.name = 'Marketing'

AND MONTH(jp.postedDate) = 1

AND YEAR(jp.postedDate) = 2011;



**Q3: Employees with no supervisees**

SELECT e1.employee\_id, p.name

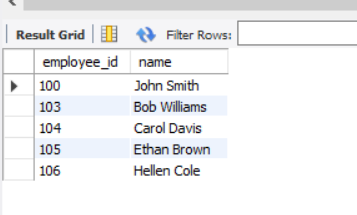
FROM Employee e1

JOIN Person p ON e1.person\_id = p.person\_id

WHERE e1.employee\_id NOT IN (

SELECT e2.supervised\_by FROM Employee e2 WHERE e2.supervised\_by IS NOT NULL

);



**Q4: Sites with no sales in March 2011**

SELECT ms.siteID, ms.location

FROM Marketing\_Site ms

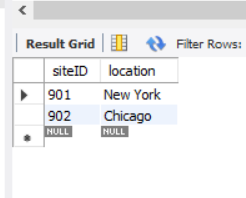
WHERE ms.siteID NOT IN (

SELECT DISTINCT siteID

FROM Sales\_History

WHERE MONTH(sale\_date) = 3 AND YEAR(sale\_date) = 2011

);



**Q5: Jobs with no hires one month after posting**

SELECT jp.jobID, jp.description

FROM Job\_Position jp

WHERE jp.jobID NOT IN (

SELECT ja.jobID

FROM Job\_Application ja

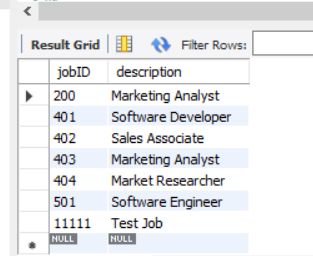
JOIN Interview i ON ja.applicationID = i.applicationID

GROUP BY ja.jobID

HAVING AVG(CAST(i.grade AS UNSIGNED)) > 70 AND COUNT(\*) >= 5

)

AND jp.postedDate < DATE\_SUB(CURDATE(), INTERVAL 1 MONTH);

\

**Q6: Salesmen who sold all product types > $200**

SELECT DISTINCT e.employee\_id, p.name

FROM Employee e

JOIN Person p ON e.person\_id = p.person\_id

WHERE NOT EXISTS (

SELECT pt.productType

FROM Product pt

WHERE pt.listPrice > 200

AND NOT EXISTS (

SELECT p2.productType

FROM Sales\_History sh

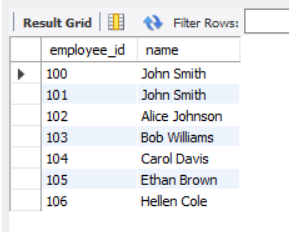
JOIN Sale\_Product sp ON sh.sale\_id = sp.sale\_id

JOIN Product p2 ON sp.product\_id = p2.product\_id

WHERE sh.employee\_id = e.employee\_id AND p2.listPrice > 200

)

);



**Q7: Departments with no job posts in Jan–Feb 2011**

SELECT d.departmentID, d.name

FROM Department d

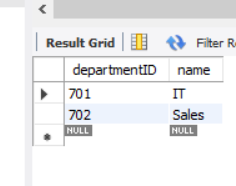
WHERE d.departmentID NOT IN (

SELECT departmentiD

FROM Job\_Position

WHERE postedDate BETWEEN '2011-01-01' AND '2011-02-01'

);



**Q8: Existing employees who applied for job “12345”**

SELECT DISTINCT p.name, e.employee\_id, ed.departmentID

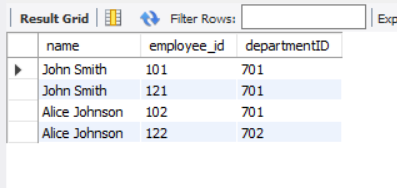
FROM Employee e

JOIN Person p ON e.person\_id = p.person\_id

JOIN Job\_Application ja ON ja.person\_id = p.person\_id

JOIN Employee\_Department ed ON e.employee\_id = ed.employee\_id

WHERE ja.jobID = 12345;

****

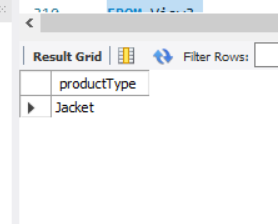
**Q9: Best selling product type (most items sold)**

SELECT productType

FROM View3

ORDER BY total\_items\_sold DESC

LIMIT 1;



**Q10: Product type with highest profit**

SELECT

p.productType,

(SUM(sp.quantity \* sp.unitPrice) - IFNULL(MAX(v.total\_part\_cost), 0)) AS profit

FROM Sale\_Product sp

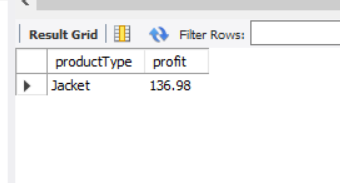
JOIN Product p ON sp.product\_id = p.product\_id

LEFT JOIN View4 v ON v.product\_id = p.product\_id

GROUP BY p.productType

ORDER BY profit DESC

LIMIT 1;



**Q11: Employees who worked in all departments**

SELECT e.employee\_id, p.name

FROM Employee e

JOIN Person p ON e.person\_id = p.person\_id

WHERE NOT EXISTS (

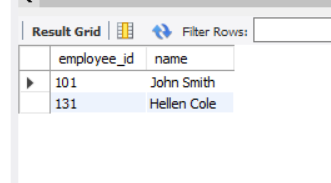
SELECT 1

FROM Department d

LEFT JOIN Employee\_Department ed ON ed.departmentID = d.departmentID AND ed.employee\_id = e.employee\_id

WHERE ed.employee\_id IS NULL

);

****

**Q12: Interviewees who are selected**

SELECT DISTINCT p.name, p.person\_id

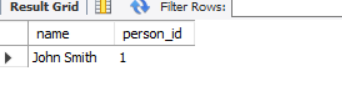
FROM Interview i

JOIN Job\_Application ja ON i.applicationID = ja.applicationID

JOIN Person p ON ja.person\_id = p.person\_id

GROUP BY p.person\_id

HAVING AVG(CAST(i.grade AS UNSIGNED)) > 70 AND COUNT(\*) >= 5;



**Q 13. Query for: "name, phone number, email of selected interviewees for all jobs they applied for"**

SELECT

per.name,

ph.phone\_number,

per.email

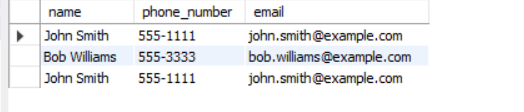
FROM Interview i

JOIN Job\_Application ja ON i.applicationID = ja.applicationID

JOIN Person per ON ja.person\_id = per.person\_id

LEFT JOIN Phone ph ON per.person\_id = ph.person\_id

WHERE i.grade IN ('A', 'A+');



**Q14: Employee with highest average monthly salary**

SELECT e.employee\_id, p.name, AVG(s.amount) AS avg\_salary

FROM Employee\_Salary s

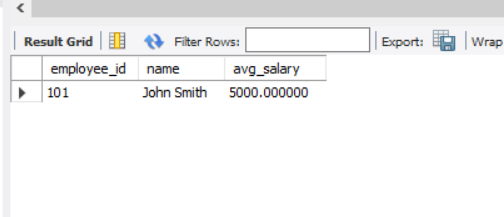
JOIN Employee e ON s.employee\_id = e.employee\_id

JOIN Person p ON e.person\_id = p.person\_id

GROUP BY e.employee\_id

ORDER BY avg\_salary DESC

LIMIT 1;



**Q15: Vendor who supplies “Cup” < 4 lbs at lowest price**

SELECT v.vendorID, v.name

FROM Vendor v

JOIN Vendor\_Part vp ON v.vendorID = vp.vendorID

JOIN Part p ON vp.part\_id = p.part\_id

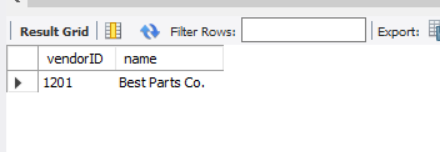
WHERE p.partType = 'Cup' AND p.price = (

SELECT MIN(p2.price)

FROM Part p2

WHERE p2.partType = 'Cup' AND p2.price < 4

);

****

**Q16: 5 Likely Business Rules**

1. A person can be an employee, customer, or both.
2. Each interview must be conducted by at least one interviewer.
3. A job application must belong to an existing job position and person.
4. Every employee must belong to at least one department.
5. Products are made up of one or more parts sourced from vendors.

**Q17: Is subclass/superclass modeling important here?**

Yes. The Person entity is a clear superclass, while Employee, Customer, and Potential\_Employee are subclasses. Using subclass/superclass (a.k.a. generalization/specialization) helps avoid redundancy and ensures integrity when different roles share common attributes (like name, address).

**Q18: Why is a relational DBMS like Oracle suitable?**

* Strong support for referential integrity via foreign keys.
* Complex queries using joins and views.
* Support for stored procedures and triggers to enforce business logic.
* Scalability and concurrency control for large enterprise use.
* Mature ecosystem with backup, security, and analytics tools.