DBMS Project

Short Summary for Warehouse System

B.Tech(CS) Honours

(2023-24)

Submitted By: -

- YASH JAIN
- AYUSH PONIA

Submitted to: -

Dr. Neeraj Gupta
HOD, B.Tech(Hons.)

DECLARATION

We ***Yash Jain, B,Tech(CS) Hons II year, 2215800033, Ayush Ponia, B,Tech(CS) Hons II year, 2215800004,*** hereby declare that the work presented in this project report entitled **Short Summary for Warehouse System** is an authentic record of our own work carried out under supervision of Dr. Neeraj Gupta HOD, B.Tech(Hons.).

YASH JAIN, 2215800033 **AYUSH PONIA, 2215800004**

CERTIFICATE

This is to certify that the above statement made by the students are correct to the best of my knowledge and belief.
Date:

Place: Mathura

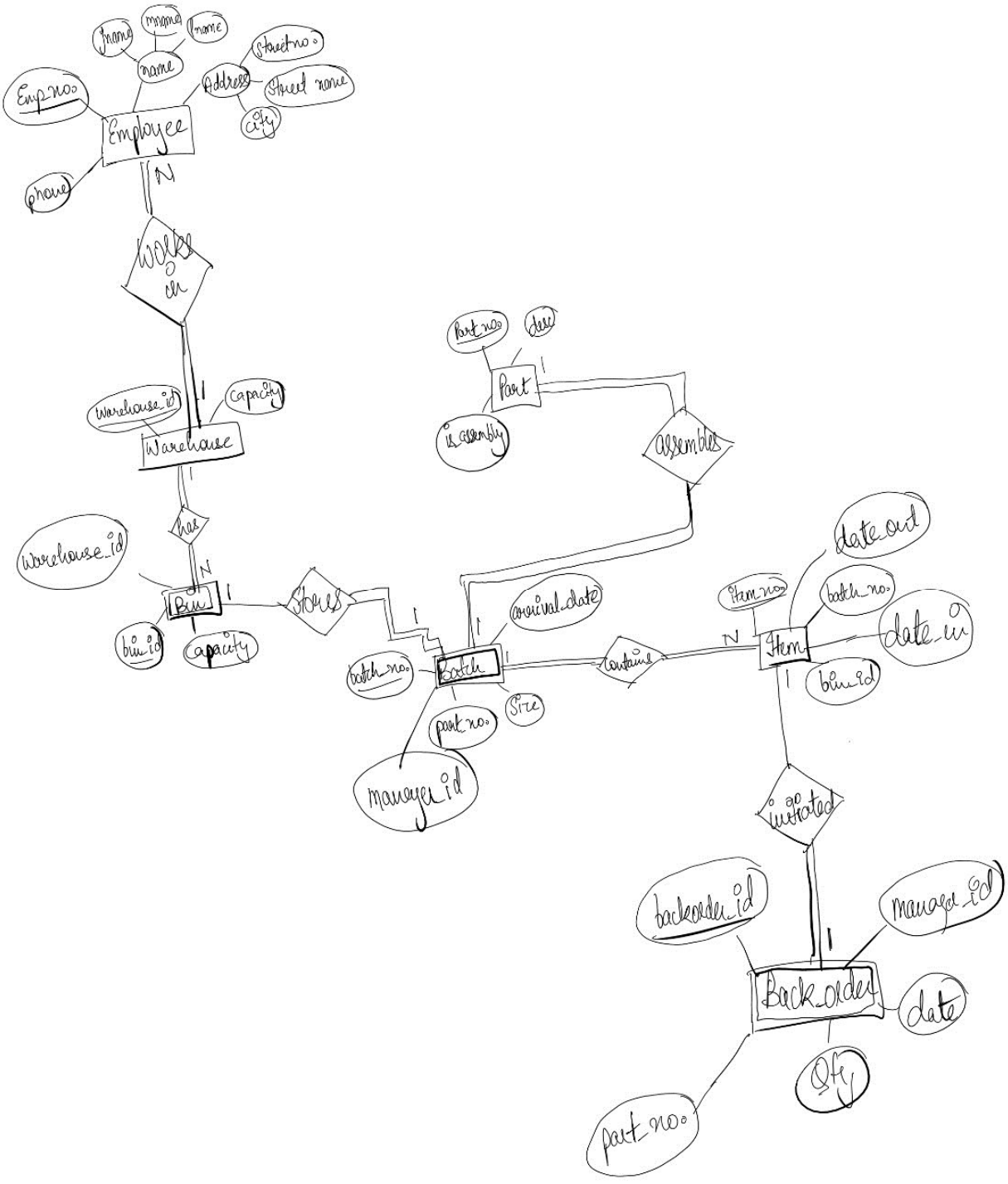
Name and Signature with Affiliation of Supervisor:

Contents

Problem Statement

Our company has a number of warehouses, and each one is identified by a distinct four- letter symbol (by letter, we mean a..z and A..Z). Each warehouse has a number of bins that are uniquely recognized by numbers (unsigned integers), such as bins 0, 1, 2, 3,... Each container has a specific capacity. We keep parts in our warehouses, or more properly, in the bins in our warehouses. Each part is identified by a specific part number, which is a five-symbol combination of digits and letters. A part can be created by joining several pieces together. This kind of component is referred to as "assembly". We just keep the individual components in the warehouses, but we record the assemblies in our database asif they were individual components. Assemblies cannot be parts of other assemblies. A part can be a constituent part in at most in one assembly parts arrive in batches. Each batch for a particular part has a unique batch number (unsigned integer) and arrives on a particular date. Each batch has a size, i.e. the number of items in the batch. All items from the same batch are stored together in the same bin (no batch is stored in more than 1 bin). Each itemin a batch has a unique item number (unsigned integer). For example: part A1, batch 27, item 1 or part A1, batch 23, item 1 etc. A specific manager must validate a batch's arrival and the database must reflect this information before the batch's date-in can be logged. Several parts might be on backorder. Only a manager has the authority to backordera part. The manager, the date of the backorder are recorded, and also the quantity backordered. When a backorder shipment arrives, the backorder's remaining quantity is updated (the number of items arrived is subtracted from the remaining quantity), and if it is less or equal to 0, the backorder is deleted, but must be kept for record. There may be only a single current (active) backorder for any parts. Assemblies cannot be backordered, only their constituent parts. When an item departs the warehouse, the employee who verified its shipping andthe date-out is also noted. Employee has a unique employee number (a 6-digit number), phone number(s) (it Case Study BCSC1003: Database Management System consists of a 3-digit area code and a 6-digit number an employee can have 0 to many phone numbers), name(s) (it consists of an up=to-10-characters fist name, an up-to-10- characters middle name, and an up-to-20-characters last name, an employee can have 1 tomany names), address(s) (it consists of an up-to-6-characters street number, an up-to20- characters street name, an up-to-20-characters city name, and a 2-character abbreviation of the province, an employee can have 1 to many address). There are managers among the staff members. One manager is responsible for overseeing all employees who are not managers. Managers are not subordinates to other managers. For the following case study in order to get the required outcome follow below given necessary steps:

(i). Draw the ER diagram for below given description



(ii). Convert the ER diagram into Relational Database.

Warehouse

Column Name	Data Type	Description
warehouse_id	INT	Unique identifier for the warehouse
capacity	INT	Tells the capacity of the warehouse

Bin

Column Name	Data Type	Description
bin_id	INT	Unique identifier for the bin
warehouse_id	VARCHAR(4)	Foreign key to the Warehouse table
capacity	INT	Capacity of the bin

Part

Column Name	Data Type	Description
part_no	VARCHAR(5)	Unique five-character part number
description	VARCHAR(50)	Description of the part
is_assembly	INT	Part is in assembly or not

Batch

Column Name	Data Type	Description
batch_number	INT	Unique identifier for the batch
part_number	VARCHAR(50)	Foreign key to the Part table or Assembly table
arrival_date	DATE	Date that the batch arrived at the warehouse
sizeB	INT	Number of parts or assemblies in the batch
manager_confirmation	INT	Manager confirmed or not

Item

Column Name	Data Type	Description
item_number	INT	Unique identifier for the item
batch_number	INT	Foreign key to the Batch table
date_in	DATE	Date that the item arrived at the warehouse
date_out	DATE	Date that the item was shipped out of the warehouse

Backorder

Column Name	Data Type	Description
backorder_id	INT	Unique identifier for the backorder
part_number	VARCHAR(5)	Foreign key to the Part table or Assembly table
manager_id	INT	Id of manager who received backorder
Backorder_date	DATE	Date of backorder placing
Quantity_backordered	INT	Quantity of items backordered
Remaining_quantity	INT	Quantity which remains after backorder

Employee

Column Name	Data Type	Description
employee_number	INT	Unique six-digit employee number
first_name	VARCHAR(10)	First name of employee
middle_name	VARCHAR(10)	Middle name of employee
last_name	VARCHAR(10)	Last name of employee
manager_no	INT	Foreign key to the Employee table (optional)

Phone_Number

Column Name	Data Type	Description
employee_number	INT	Foreign key to the employee table
Phone_number	VARCHAR(10)	Phone number of employee

Address

Column Name	Data Type	Description
employee_number	INT	Foreign key to employee table
street_number	VARCHAR(4)	Street number where employee lives
Street_name	VARCHAR(20)	Street name where employee lives
city	VARCHAR(20)	City where employee lives
province	VARCHAR(20)	Province where employee lives

(iii). Remove any functional dependencies (if any).

Table: Warehouse

- Warehouse_id: Primary key, unique identifier for the warehouse
- Capacity: Tells the capacity of the warehouse

No functional dependencies identified.

Table: Bin

- Bin_id: Primary key, unique identifier for the bin
- Warehouse_id: Foreign key to the warehouse table
- Capacity: capacity of the bin

No functional dependencies identified.

Table: Part

- Part_no: Primary key, unique five-character part number
- Description: description of part
- Is_assembly: part is in assembly or not

No functional dependencies identified.

Table: Batch

- Batch_number: Primary key, unique identifier for batch
- Part_number: foreign key to part table or assembly table
- Arrival_date: date batch arrived at warehouse
- sizeB: number of parts or assemblies in batch
- manager_confirmation: manager confirmed or not

No functional dependencies identified.

Table: Item

- item_number: Primary key, unique identifier for item
- batch_number: foreign key to batch table
- bin_id: foreign key to bin table
- date_out: date when item was shipped out of warehouse
- checked_by_employee_number: employee checked or not

No functional dependencies identified.

Table: Backorder

- backorder_id: Primary key, unique identifier for backorder
- part_number: foreign key to the part or assembly table
- manager_id: id of manager who received backorder
- backorder_date: date of backorder placing
- quantity_backordered: quantity of items backordered
- remaining_quantity: quantity which remains after backorder
- is_current: its current scenario

No functional dependencies identified.

Table: Employee

- employee_number: Primary key, unique identifier for employee
- first_name: first name of employee
- middle_name: middle name of employee
- last_name: last name of employee
- is_manager: has manager or not
- manager_id: foreign key to the employee table

No functional dependencies identified.

Table: Phone_Number

- employee_number: foreign key to the employee table
- phone_number: phone number of employees

No functional dependencies identified.

Table: Address

- employee_number: foreign key to employee table
- street_number: street number where employee lives
- street_name: street name where employee lives
- city: city where employee lives
- province: province where employee lives

No functional dependencies identified.

(iv). Implement in Oracle Live SQL and give the required outcome as query results.

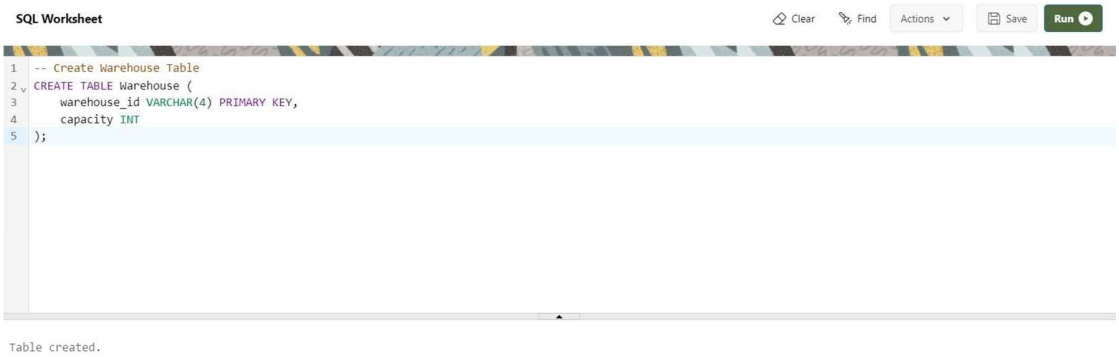


Table created.

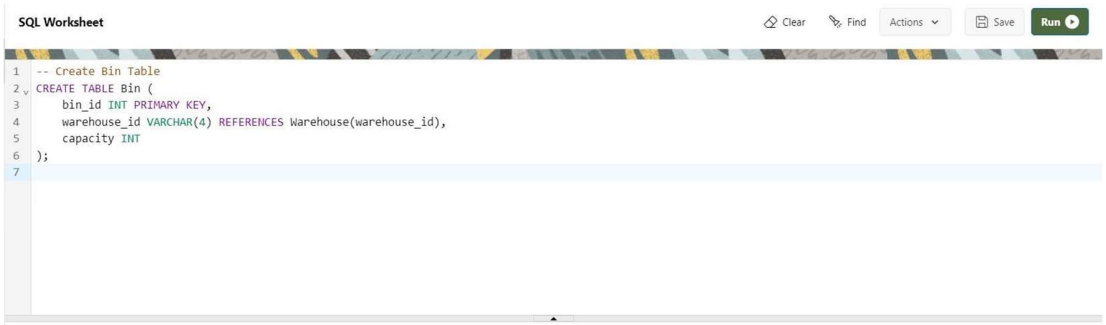


Table created.

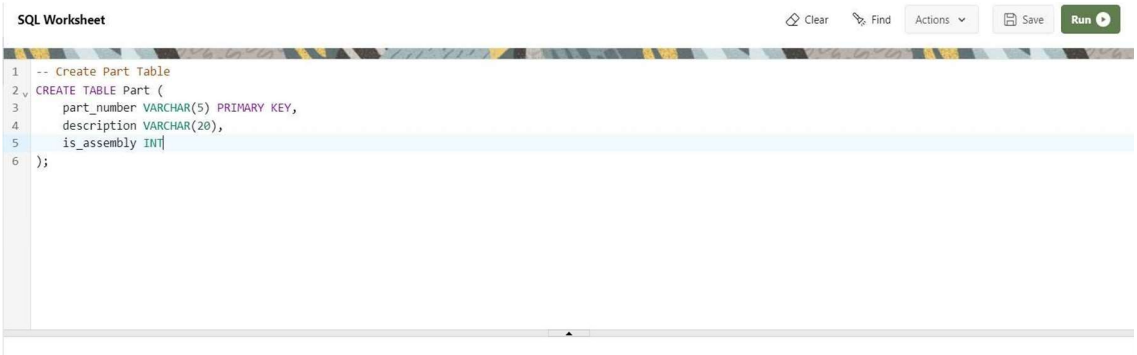


Table created.

SQL Worksheet

Clear Find Actions Save Run

```sql
-- Create Batch Table
CREATE TABLE Batch (
    batch_number INT PRIMARY KEY,
    part_number VARCHAR(5) REFERENCES Part(part_number),
    arrival_date DATE,
    sizeB INT,
    manager_id INT REFERENCES Employee(employee_number)
);

```

```
Table created.
```

SQL Worksheet

Clear Find Actions Save Run

```sql
-- Create Item Table
CREATE TABLE Item (
    item_number INT PRIMARY KEY,
    batch_number INT REFERENCES Batch(batch_number),
    bin_id INT REFERENCES Bin(bin_id)
);
```

```
Table created.
```

SQL Worksheet

Clear Find Actions Save Run

```sql
-- Backorder Table
CREATE TABLE Backorder (
    backorder_id INT PRIMARY KEY,
    part_number VARCHAR(5) REFERENCES Part(part_number),
    manager_id INT,
    backorder_date DATE,
    quantity_backordered INT,
    remaining_quantity INT,
    is_current INT
);

```

```
Table created.
```

SQL Worksheet

Clear Find Actions Save Run

```sql
-- Create Phone_Number Table
CREATE TABLE Phone_Number (
    employee_number INT REFERENCES Employee(employee_number),
    phone_number VARCHAR(10),
    PRIMARY KEY (employee_number, phone_number)
);
```

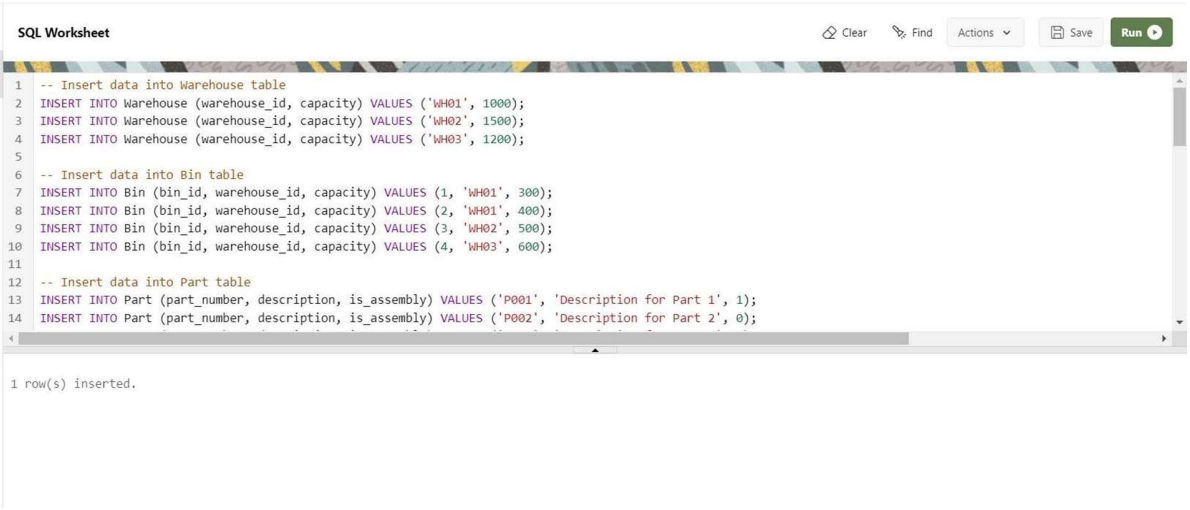
```
Table created.
```

SQL Worksheet

Clear Find Actions Save Run

```sql
-- Create Address Table
CREATE TABLE Address (
    employee_number INT REFERENCES Employee(employee_number),
    street_number VARCHAR(6),
    street_name VARCHAR(20),
    city VARCHAR(20),
    province VARCHAR(2),
    PRIMARY KEY (employee_number)
);
```

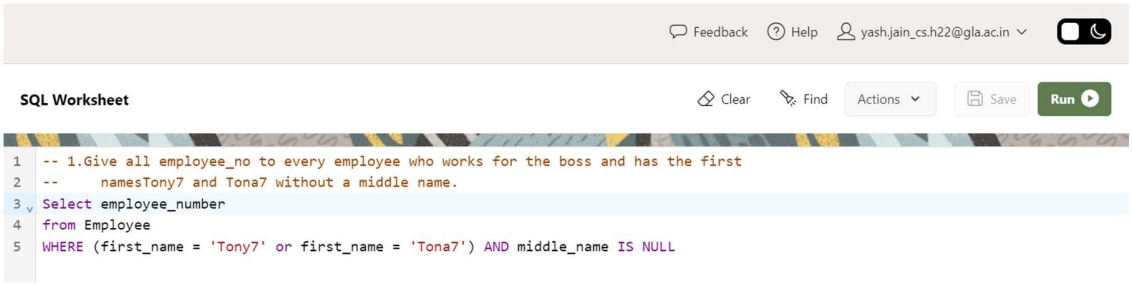
```
Table created.
```



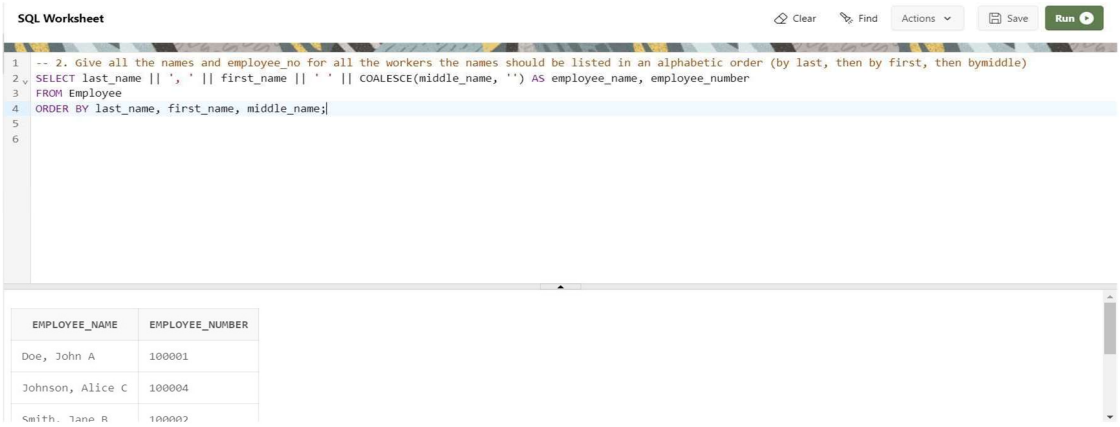

```
-- Insert data into Warehouse table
INSERT INTO Warehouse (warehouse_id, capacity) VALUES ('WH01', 1000);
INSERT INTO Warehouse (warehouse_id, capacity) VALUES ('WH02', 1500);
INSERT INTO Warehouse (warehouse_id, capacity) VALUES ('WH03', 1200);

-- Insert data into Bin table
INSERT INTO Bin (bin_id, warehouse_id, capacity) VALUES (1, 'WH01', 300);
INSERT INTO Bin (bin_id, warehouse_id, capacity) VALUES (2, 'WH01', 400);
INSERT INTO Bin (bin_id, warehouse_id, capacity) VALUES (3, 'WH02', 500);
INSERT INTO Bin (bin_id, warehouse_id, capacity) VALUES (4, 'WH03', 600);

-- Insert data into Part table
INSERT INTO Part (part_number, description, is_assembly) VALUES ('P001', 'Description for Part 1', 1);
INSERT INTO Part (part_number, description, is_assembly) VALUES ('P002', 'Description for Part 2', 0);
INSERT INTO Part (part_number, description, is_assembly) VALUES ('P003', 'Description for Part 3', 1);
INSERT INTO Part (part_number, description, is_assembly) VALUES ('P004', 'Description for Part 4', 0);

-- Insert data into Batch table
INSERT INTO Batch (batch_number, part_number, arrival_date, sizeb, manager_confirmation) VALUES (1, 'P001',
TO_DATE('2023-11-20', 'YYYY-MM-DD'), 50, 1);
INSERT INTO Batch (batch_number, part_number, arrival_date, sizeb, manager_confirmation) VALUES (2, 'P002',
TO_DATE('2023-11-21', 'YYYY-MM-DD'), 30, 1);
INSERT INTO Batch (batch_number, part_number, arrival_date, sizeb, manager_confirmation) VALUES (3, 'P003',
TO_DATE('2023-11-22', 'YYYY-MM-DD'), 40, 0);

-- Insert data into Item table
INSERT INTO Item (item_number, batch_number, bin_id, date_out, checked_by_employee_number) VALUES (1, 1, 1,
TO_DATE('2023-11-25', 'YYYY-MM-DD'), 100001);
```

```
INSERT INTO Item (item_number, batch_number, bin_id, date_out, checked_by_employee_number) VALUES (2, 1, 2, TO_DATE('2023-11-26', 'YYYY-MM-DD'), 100002);

INSERT INTO Item (item_number, batch_number, bin_id, date_out, checked_by_employee_number) VALUES (3, 2, 3, TO_DATE('2023-11-24', 'YYYY-MM-DD'), 100003);

INSERT INTO Item (item_number, batch_number, bin_id, date_out, checked_by_employee_number) VALUES (4, 3, 4, NULL, NULL);


-- Insert data into Backorder table

INSERT INTO Backorder (backorder_id, part_number, manager_id, backorder_date, quantity_backordered, remaining_quantity, is_current) VALUES (1, 'P002', 100003, TO_DATE('2023-11-23', 'YYYY-MM-DD'), 20, 20, 1);

INSERT INTO Backorder (backorder_id, part_number, manager_id, backorder_date, quantity_backordered, remaining_quantity, is_current) VALUES (2, 'P004', 100003, TO_DATE('2023-11-24', 'YYYY-MM-DD'), 15, 15, 1);

INSERT INTO Backorder (backorder_id, part_number, manager_id, backorder_date, quantity_backordered, remaining_quantity, is_current) VALUES (3, 'P001', 100004, TO_DATE('2023-11-25', 'YYYY-MM-DD'), 10, 10, 1);


-- Insert data into Employee table

INSERT INTO Employee (employee_number, first_name, middle_name, last_name, is_manager, manager_id) VALUES (100001, 'John', 'A', 'Doe', 1, NULL);

INSERT INTO Employee (employee_number, first_name, middle_name, last_name, is_manager, manager_id) VALUES (100002, 'Jane', 'B', 'Smith', 0, 100001);

INSERT INTO Employee (employee_number, first_name, middle_name, last_name, is_manager, manager_id) VALUES (100003, 'Tony7', NULL, 'Tona7', 1, 100003);

INSERT INTO Employee (employee_number, first_name, middle_name, last_name, is_manager, manager_id) VALUES (100004, 'Alice', 'C', 'Johnson', 0, 100001);

INSERT INTO Employee (employee_number, first_name, middle_name, last_name, is_manager, manager_id) VALUES (100005, 'Bob', 'D', 'Williams', 0, 100001);


-- Insert data into Phone_Number table

INSERT INTO Phone_Number (employee_number, phone_number) VALUES (100001, '1234567890');

INSERT INTO Phone_Number (employee_number, phone_number) VALUES (100002, '9876543210');

INSERT INTO Phone_Number (employee_number, phone_number) VALUES (100003, '5551112222');

INSERT INTO Phone_Number (employee_number, phone_number) VALUES (100004, '9998887777');


-- Insert data into Address table

INSERT INTO Address (employee_number, street_number, street_name, city, province) VALUES (100001, '123', 'Main St', 'Cityville', 'AB');

INSERT INTO Address (employee_number, street_number, street_name, city, province) VALUES (100002, '456', 'Oak St', 'Townsville', 'BC');

INSERT INTO Address (employee_number, street_number, street_name, city, province) VALUES (100003, '789', 'Maple St', 'Villageton', 'ON');

INSERT INTO Address (employee_number, street_number, street_name, city, province) VALUES (100004, '101', 'Cedar St', 'Hamletown', 'QC');
```

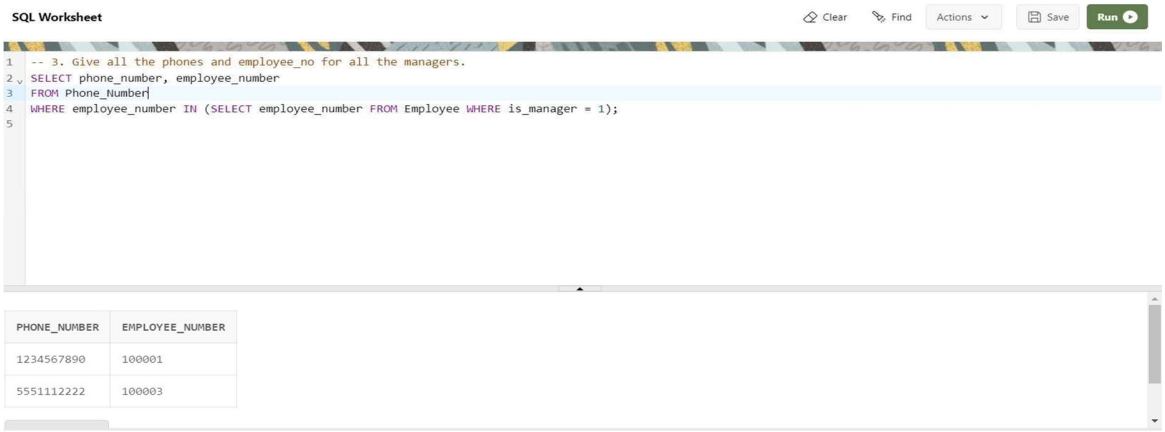
1. Give all employee_no to every employee who works for the boss and has the first namesTony7 and Tona7 without a middle name.



2. Give all the names and employee_no for all the workers the names should be listed in an alphabetic order (by last, then by first, then by middle)



3. Give all the phones and employee_no for all the managers.



4. List all parts that are assemblies they should be listed in a lexicographic order.

SQL Worksheet

Clear Find Actions Save Run

```
-- 4. List all parts that are assemblies they should be listed in a lexicographicorder.
SELECT part_number FROM Part
WHERE is_assembly = 1
ORDER BY part_number;
```

PART_NUMBER
P001
P003

5. For each manager, list all current backorders done by the manager.

SQL Worksheet

Clear Find Actions Save Run

```
-- 5. For each manager, list all current backorders done by the manager.
SELECT manager_id, part_number, backorder_date, TO_DATE('2000-01-01', 'YYYY-MM-DD') AS fulfilled_date
FROM Backorder
WHERE is_current = 1;
```

MANAGER_ID	PART_NUMBER	BACKORDER_DATE	FULFILLED_DATE
100003	P002	23-NOV-23	01-JAN-00
100003	P004	24-NOV-23	01-JAN-00
100004	P001	25-NOV-23	01-JAN-00

6. For each manager, list all current and old backorders done by the manager. For each backorder you have to list the part_no, backorder date, and fulfilled date. For current backorders, list a phony fulfilled date '2000-01-01'.

SQL Worksheet

Clear Find Actions Save Run

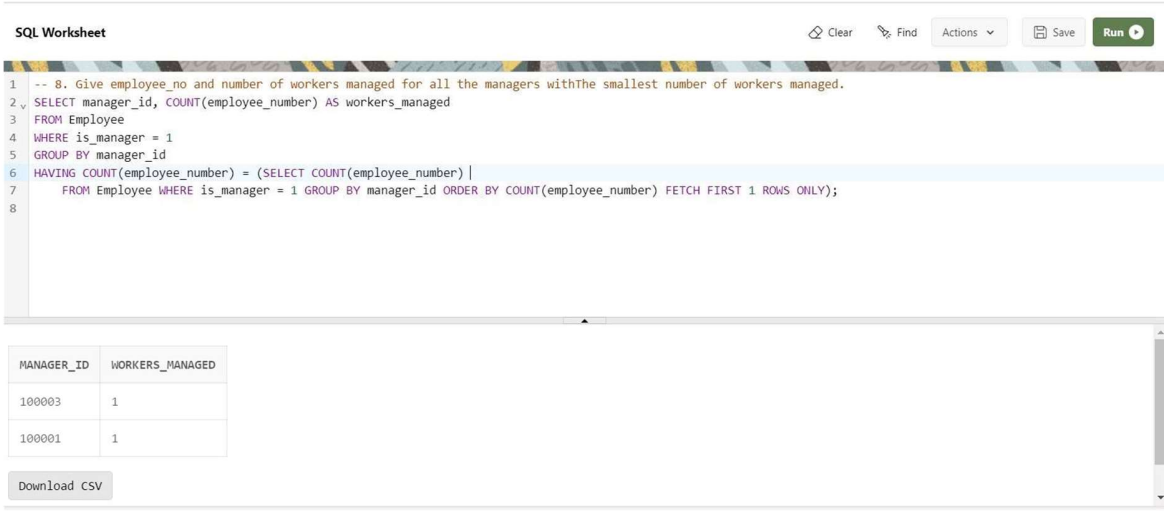
```
-- 6. For each manager, list all current and old backorders done by the manager. For each backorder you have to list the part_no,
-- backorder date, and fulfilled date. For current backorders, list a phony fulfilled date '2000-01-01'.
SELECT manager_id, part_number, backorder_date, TO_DATE('2000-01-01', 'YYYY-MM-DD') AS fulfilled_date
FROM Backorder;
```

MANAGER_ID	PART_NUMBER	BACKORDER_DATE	FULFILLED_DATE
100003	P002	23-NOV-23	01-JAN-00
100003	P004	24-NOV-23	01-JAN-00
100004	P001	25-NOV-23	01-JAN-00

7. For each warehouse bin, give the remaining capacity of the bin. Call the remaining capacity remaining_capacity.

BIN_ID	REMAINING_CAPACITY
3	499
2	399
4	599

8. Give employee_no and number of workers managed for all the managers with The smallest number of workers managed



MANAGER_ID	WORKERS_MANAGED
100003	1
100001	1

Download CSV