AMERICAN INTERNATIONAL UNIVERSITY–BANGLADESH

(AIUB)

Introduction to Database

Spring 2021-2022

Database Final Project

Supervised By: Dr. S.M. HASAN Mahmud

Topic: Food Factory Management System

Section: G

Submitted by:

Name	ID
Ashik Ahamed	21-45368-2
Srabone Raxit	21-45038-2
Irtiza Ahsan Abir	21-45009-2
Khandaker Mahadi Ahmed	21-44683-1

1. Introduction:

Food is the essential needs in our life. After observing many resources, we choose to do our project in "Food Factory Management System". We found it useful as people don't know how a food factory works internally. Our purpose is to make people know about the internal management system. We are only in our learning stage, and we tried our best to use as much as information which we learnt these days in our "Introduction to Database" program.

2. Content List:

3. Scenario:

In our "Food Factory Management System" we include *worker, stock, food, customer, visitor, machine, and records*. These things are collected directly. As we try to visualize it with reality, we build relationship between these things. These are known as entity in database. We also include connection between those entities and named the connection according to our thoughts and implementation plans.

4. ER-Diagram:

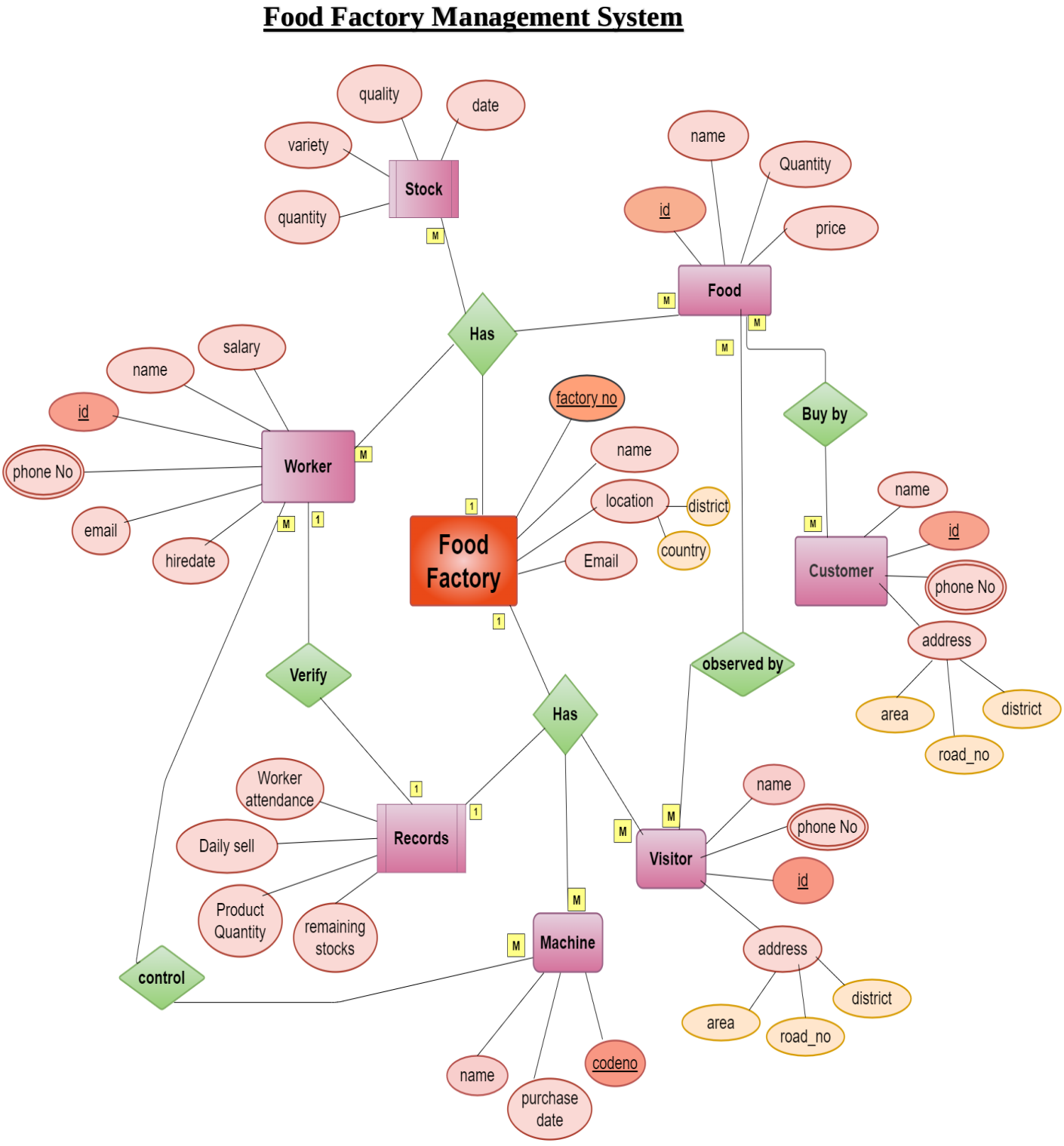
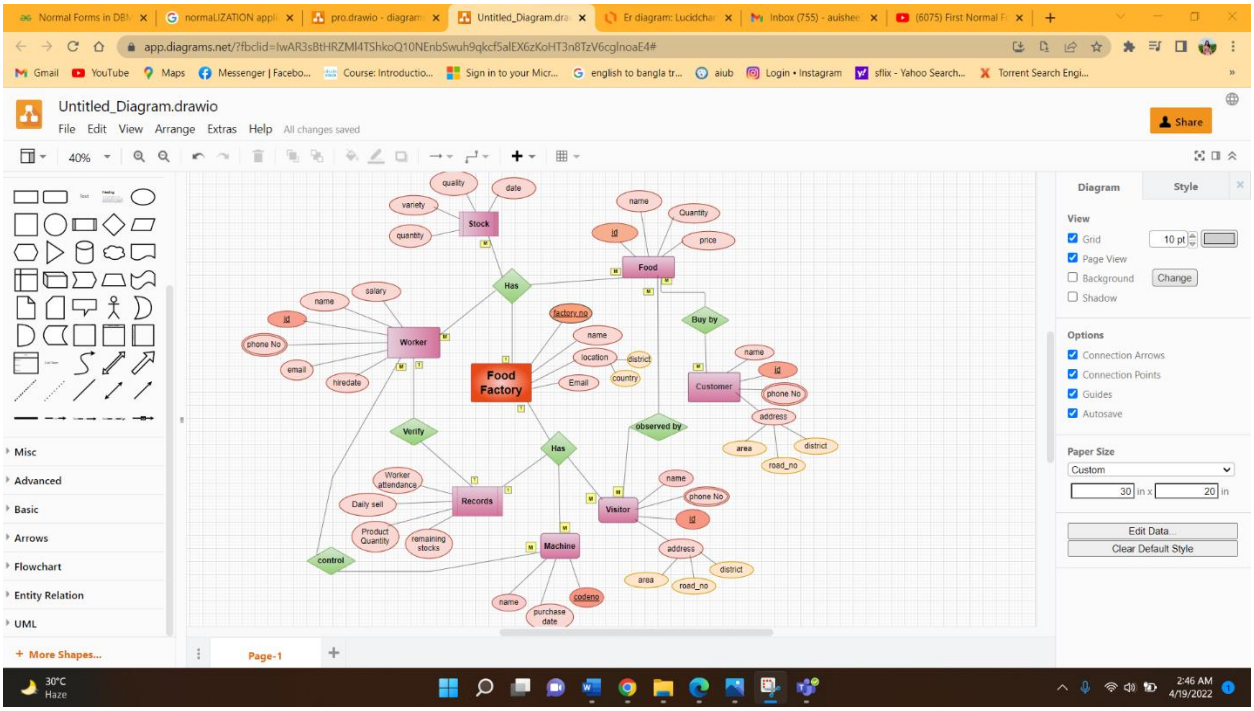


Figure: ER-Diagram on "Food Factory Management System"

5. Explanation:



Reason for including the screenshot is to let our faculty know that we do the project by ourselves. If he concerns, he can verify it by checking date & time.

In our diagram, we added some colors to look the diagram more beautiful and so that we can see the different colors and quickly understand which one define entities, which one define attributes, which one define relationships etc. To define the relationship types between entities we use ‘M’ for many and ‘1’ for one. We can easily understand for these notations that which entities represent what type of relationship between them. In our diagram, we use 4 types of relationship between the entities. These are-

- One to one relationship
- One to many relationship
- Many to one relationship
- Many to many relationship

There is total 8 entities. We include relationship between *food factory, worker, stock, food, customer, visitor, machine, and records* which are acting as entities. Here, rectangle shapes

represent the entities, ellipse shapes represent the attributes of the entities, and the diamond shapes represents the relationship between two entities.

Entities, attributes, and the types of the attributes:

We use diagrams.net to make this ER diagram. The table shows the entity name with its attributes and its' attribute types.

Entity Name	Attributes	Attribute Types
Food Factory	• factory_no • name, email • location	• Primary • Simple • Composite
Worker	• ID • Phone_no • Salary, name, email, hiredate	• Primary • Multi-valued • simple
Stock	• quality, quantity, variety, date	• simple
Food	• ID • Name, quantity, price	• Primary • Simple
Customer	• ID • Phone_no • Address • name	• Primary • Multi-valued • Composite • simple
Visitor	• ID • Phone_no • Address • name	• Primary • Multi-valued • Composite • Simple

Machine	• code_no • purchase_date, name	• Primary • Simple
Records	• worker_attendence, daily_sell, product_quantity, remaining_stocks	• Simple

6. Normalization:

Normalization is a database design technique that reduces data redundancy and eliminates undesirable characteristics like insertion, update, and deletion anomalies. Normalization rules divides larger tables into smaller tables and links them using relationships.

I. 1NF:

First Normal Form does not contain composite or multivalued attributes.

Example:

Table ***VISITOR***:

Before 1NF:

ID	NAME	PHONE_NO	ADDRESS
101	NAHID	0181xxxxx	GULSHAN,DHAKA,3/D
102	SHAMIM	0181xxxxx	RAMPURA,DHAKA,20/G
103	SHAWN	0181xxxxx	BASHABO,DHAKA,1/5
104	AKRAM	0181xxxxx	NORDA,DHAKA,5/E

4 rows returned in 0.00 seconds CSV Export

After 1NF:

Results Explain Describe Saved SQL History

ID	NAME	PHONE_NO	AREA	DISTRICT	ROAD_NO
101	NAHID	0181xxxxx	GULSHAN	DHAKA	3/D
102	SHAMIM	0181xxxxx	RAMPURA	DHAKA	20/G
103	SHAWN	0181xxxxx	BASHABO	DHAKA	1/5
104	AKRAM	0181xxxxx	NORDA	DHAKA	5/E

4 rows returned in 0.00 seconds CSV Export

- **Table creation:**

```
CREATE TABLE VISITOR_1F
(
ID VARCHAR (20) PRIMARY KEY,
NAME VARCHAR (30) NOT NULL,
PHONE_NO VARCHAR (30),
AREA VARCHAR (50),
DISTRICT VARCHAR (50),
ROAD_NO VARCHAR (50)
);
```

- **Values:**

```
INSERT INTO VISITOR_1F
VALUES
(101,'NAHID','0181xxxxx','GULSHAN', 'DHAKA','3/D');

INSERT INTO VISITOR_1F
VALUES
(102,'SHAMIM','0181xxxxx','RAMPURA', 'DHAKA','20/G');
```

```
INSERT INTO VISITOR_1F
VALUES
(103,'SHAWN','0181xxxxx','BASHABO', 'DHAKA','1/5');

INSERT INTO VISITOR_1F
VALUES
(104,'AKRAM','0181xxxxx','NORDA', 'DHAKA','5/E');
```

II. 2NF

To be Second Normal Form, a relation must be in 1NF, and relation must not contain any partial dependency.

Example:

Table ***WORKER***

Before 2NF:

The table WORKER is already in 1NF form.

ID	NAME	SALARY	PHONE_NO	EMAIL	HIREDATE
1	kamal	2000	0181xxxxx	kamal@gmaiL.com	1-1-2017
3	karim	4500	0181xxxxx	karim@gmail.com	7-7-2018
2	jamal	4000	0181xxxxx	jamal@gmail.com	3-7-2017
4	Rahim	5000	0181xxxxx	Rahim@gmail.com	2-2-2019

4 rows returned in 0.02 seconds CSV Export

After 2NF:

- Table 1: Worker's information in terms of public view

ID	NAME	PHONE_NO
1	kamal	0181xxxxx
3	karim	0181xxxxx
2	jamal	0181xxxxx
4	Rahim	0181xxxxx

4 rows returned in 0.00 seconds CSV Export

- Table 2: Worker's information in terms of BOSS view

ID	SALARY	EMAIL	HIREDATE
1	2000	kamal@gmail.com	1-1-2017
3	4500	karim@gmail.com	7-7-2018
2	4000	jamal@gmail.com	3-7-2017
4	5000	Rahim@gmail.com	2-2-2019

4 rows returned in 0.00 seconds CSV Export

Table 1 & 2 creation & Value Insertion

Table 1:

- **Creation:**

```
CREATE TABLE WORKER_2NF1
(
ID VARCHAR (20) PRIMARY KEY,
NAME VARCHAR (30) NOT NULL,
PHONE_NO VARCHAR (30)
);
```

- **Values:**

```
INSERT INTO WORKER_2NF1
VALUES (1, 'kamal', '0181xxxxx');

INSERT INTO WORKER_2NF1
VALUES (3, 'karim', '0181xxxxx');

INSERT INTO WORKER_2NF1
VALUES (2, 'jamal', '0181xxxxx');

INSERT INTO WORKER_2NF1
VALUES (4, 'Rahim', '0181xxxxx');
```

Table 2:

- **Creation:**

```
CREATE TABLE WORKER_2NF2
(
ID SMALLINT PRIMARY KEY,
SALARY DECIMAL (6,2) NOT NULL,
EMAIL VARCHAR (50),
HIREDATE VARCHAR (40)
);
```

- **Values:**

```
INSERT INTO WORKER_2NF2
VALUES (1, 2000, 'kamal@gmail.com', '1-1-2017');

INSERT INTO WORKER_2NF2
VALUES (3, 4500, 'karim@gmail.com', '7-7-2018');

INSERT INTO WORKER_2NF2
VALUES (2, 4000, 'jamal@gmail.com', '3-7-2017');

INSERT INTO WORKER_2NF2
VALUES (4, 5000, 'Rahim@gmail.com', '2-2-2019');
```

III. 3NF

If there is no transitive dependency for non-prime attributes as well as it is in Second Normal Form.

Transitive Dependency is a functional dependency between two or more non-key attributes.

In our diagram, 3NF is not applicable.

7. Table Creation:

I. FOOD FACTORY

```
CREATE TABLE FOOD_FACTORY
(
FACTORY_NO SMALLINT PRIMARY KEY,
NAME VARCHAR (30) NOT NULL,
LOCATION VARCHAR (30),
EMAIL VARCHAR (50)
);
```

Object Type **TABLE** Object **FOOD_FACTORY**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
FOOD_FACTORY	FACTORY_NO	Number	-	-	0	1	-	-	-
	NAME	Varchar2	30	-	-	-	-	-	-
	LOCATION	Varchar2	30	-	-	-	✓	-	-
	EMAIL	Varchar2	50	-	-	-	✓	-	-

1 - 4

II. WORKER

```
CREATE TABLE WORKER
(
ID SMALLINT PRIMARY KEY,
NAME VARCHAR (30) NOT NULL,
SALARY DECIMAL (6,2) NOT NULL,
PHONE_NO VARCHAR (30),
EMAIL VARCHAR (50),
HIREDATE VARCHAR (40)
);
```

Query Plan

Operation	Options	Object	Rows	Time	Cost	Bytes	Filter Predicates *	Access Predicates
SELECT STATEMENT			1	1	2	109		
TABLE ACCESS	FULL	WORKER	1	1	2	109		

* Unindexed columns are shown in red

Index Columns

Owner	Table Name	Index Name	Used In Plan	Columns	Uniqueness	Status	Index Type	Join Index
SRABONERAXIT	WORKER	SYS_C004342		ID	UNIQUE	VALID	NORMAL	NO

Table Columns

Table Owner	Table Name	Column Name	Data Type
SRABONERAXIT	WORKER	ID	NUMBER
		NAME	VARCHAR2
		SALARY	NUMBER
		PHONE_NO	VARCHAR2
		EMAIL	VARCHAR2
		HIREDATE	VARCHAR2

Object Type **TABLE** Object **WORKER**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
WORKER	ID	Number	-	-	0	1	-	-	-
	NAME	Varchar2	30	-	-	-	-	-	-
	SALARY	Number	-	6	2	-	-	-	-
	PHONE_NO	Varchar2	30	-	-	-	✓	-	-
	EMAIL	Varchar2	50	-	-	-	✓	-	-
	HIREDATE	Varchar2	40	-	-	-	✓	-	-

1 - 6

III. STOCK

```
CREATE TABLE STOCK
(
S_VARIETY VARCHAR (30) NOT NULL,
S_QUALITY VARCHAR (40),
S_QUANTITY VARCHAR (50) NOT NULL,
S_DATE VARCHAR (40)
);
```

Table Columns

Table Owner	Table Name	Column Name	Data Type
SRABONERAXIT	STOCK	S_VARIETY	VARCHAR2
		S_QUALITY	VARCHAR2
		S_QUANTITY	VARCHAR2
		S_DATE	VARCHAR2

Object Type **TABLE** Object **STOCK**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
STOCK	S_VARIETY	Varchar2	30	-	-	-	-	-	-
	S_QUALITY	Varchar2	40	-	-	-	✓	-	-
	S_QUANTITY	Varchar2	50	-	-	-	-	-	-
	S_DATE	Varchar2	40	-	-	-	✓	-	-

1 - 4

IV. FOOD

```
(
ID VARCHAR (20) PRIMARY KEY,
NAME VARCHAR (30) NOT NULL,
QUATITY VARCHAR (30),
PRICE VARCHAR (50) NOT NULL
);
```

Query Plan

Operation	Options	Object	Rows	Time	Cost	Bytes	Filter Predicates *	Access Predicates
SELECT STATEMENT			1	1	2	73		
TABLE ACCESS	FULL	FOOD	1	1	2	73		

* Unindexed columns are shown in red

Index Columns

Owner	Table Name	Index Name	Used In Plan	Columns	Uniqueness	Status	Index Type	Join Index
SRABONERAXIT	FOOD	SYS_C004333		ID	UNIQUE	VALID	NORMAL	NO

Table Columns

Table Owner	Table Name	Column Name	Data Type
SRABONERAXIT	FOOD	ID	VARCHAR2
		NAME	VARCHAR2
		QUATITY	VARCHAR2
		PRICE	VARCHAR2

Object Type **TABLE** Object **FOOD**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
FOOD	ID	Varchar2	20	-	-	1	-	-	-
	NAME	Varchar2	30	-	-	-	-	-	-
	QUATITY	Varchar2	30	-	-	-	✓	-	-
	PRICE	Varchar2	50	-	-	-	-	-	-

1 - 4

V. CUSTOMER

```
CREATE TABLE CUSTOMER
(
ID SMALLINT PRIMARY KEY,
NAME VARCHAR (30),
PHONE_NO VARCHAR (30),
ADDRESS VARCHAR (50)
);
```

Query Plan

Operation	Options	Object	Rows	Time	Cost	Bytes	Filter Predicates *	Access Predicates
SELECT STATEMENT			1	1	2	73		
TABLE ACCESS	FULL	CUSTOMER	1	1	2	73		

* Unindexed columns are shown in red

Index Columns

Owner	Table Name	Index Name	Used In Plan	Columns	Uniqueness	Status	Index Type	Join Index
SRABONERAXIT	CUSTOMER	SYS_C004336		ID	UNIQUE	VALID	NORMAL	NO

Table Columns

Table Owner	Table Name	Column Name	Data Type
SRABONERAXIT	CUSTOMER	ID	VARCHAR2
		NAME	VARCHAR2
		PHONE_NO	VARCHAR2
		ADDRESS	VARCHAR2

Object Type **TABLE** Object **CUSTOMER**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
CUSTOMER	ID	Number	-	-	0	1	-	-	-
	NAME	Varchar2	30	-	-	-	-	-	-
	PHONE_NO	Varchar2	30	-	-	-	✓	-	-
	ADDRESS	Varchar2	50	-	-	-	✓	-	-

1 - 4

VI. VISITOR

```
CREATE TABLE VISITOR
(
ID VARCHAR (20) PRIMARY KEY,
NAME VARCHAR (30) NOT NULL,
PHONE_NO VARCHAR (30),
ADDRESS VARCHAR (50)
);
```

Query Plan

Operation	Options	Object	Rows	Time	Cost	Bytes	Filter Predicates *	Access Predicates
SELECT STATEMENT			1	1	2	73		
TABLE ACCESS	FULL	VISITOR	1	1	2	73		

* Unindexed columns are shown in red

Index Columns

Owner	Table Name	Index Name	Used In Plan	Columns	Uniqueness	Status	Index Type	Join Index
SRABONERAXIT	VISITOR	SYS_C004335		ID	UNIQUE	VALID	NORMAL	NO

Table Columns

Table Owner	Table Name	Column Name	Data Type
SRABONERAXIT	VISITOR	ID	VARCHAR2
		NAME	VARCHAR2
		PHONE_NO	VARCHAR2
		ADDRESS	VARCHAR2

Object Type **TABLE** Object **VISITOR**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
VISITOR	ID	Varchar2	20	-	-	1	-	-	-
	NAME	Varchar2	30	-	-	-	-	-	-
	PHONE_NO	Varchar2	30	-	-	-	✓	-	-
	ADDRESS	Varchar2	50	-	-	-	✓	-	-

1 - 4

VII. MACHINE

```
CREATE TABLE MACHINE
(
CODE_NO SMALLINT PRIMARY KEY,
NAME VARCHAR (30) NOT NULL,
PURCHASE_DATE VARCHAR (50)
);
```

Query Plan

Operation	Options	Object	Rows	Time	Cost	Bytes	Filter Predicates *	Access Predicates
SELECT STATEMENT			1	1	2	57		
TABLE ACCESS	FULL	MACHINE	1	1	2	57		

* Unindexed columns are shown in red

Index Columns

Owner	Table Name	Index Name	Used In Plan	Columns	Uniqueness	Status	Index Type	Join Index
SRABONERAXIT	MACHINE	SYS_C004330		CODE_NO	UNIQUE	VALID	NORMAL	NO

Table Columns

Table Owner	Table Name	Column Name	Data Type
SRABONERAXIT	MACHINE	CODE_NO	NUMBER
		NAME	VARCHAR2
		PURCHASE_DATE	VARCHAR2

Object Type **TABLE** Object **MACHINE**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MACHINE	CODE_NO	Number	-	-	0	1	-	-	-
	NAME	Varchar2	30	-	-	-	-	-	-
	PURCHASE_DATE	Varchar2	50	-	-	-	✓	-	-
									1 - 3

VIII. RECORDS

```
CREATE TABLE RECORDS
(
WORKER_ATTENDANCE VARCHAR (30),
DAILY_SELL VARCHAR (40),
PRODUCT_QUANTITY VARCHAR (50),
REMAINING_STOCKS VARCHAR (40)
);
```

Table Columns

Table Owner	Table Name	Column Name	Data Type
SRABONERAXIT	RECORDS	WORKER_ATTENDANCE	VARCHAR2
		DAILY_SELL	VARCHAR2
		PRODUCT_QUANTITY	VARCHAR2
		REMAINING_STOCKS	VARCHAR2

Object Type **TABLE** Object **RECORDS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
RECORDS	WORKER_ATTENDANCE	Varchar2	30	-	-	-	✓	-	-
	DAILY_SELL	Varchar2	40	-	-	-	✓	-	-
	PRODUCT_QUANTITY	Varchar2	50	-	-	-	✓	-	-
	REMAINING_STOCKS	Varchar2	40	-	-	-	✓	-	-

1 - 4

8. Data Insertion:

a. FOOD FACTORY

```
INSERT INTO FOOD_FACTORY
VALUES
(24151,'HASAN FOOD FACTORY','DHAKA',
'AAAA@GAMIL.COM');
```

FACTORY_NO	NAME	LOCATION	EMAIL
24151	HASAN FOOD FACTORY	DHAKA	AAAA@GAMIL.COM

1 rows returned in 0.00 seconds CSV Export

b. WORKER

INSERT INTO WORKER

VALUES

(1,'kamal','2000','0181xxxxx', 'kamal@gmaiL.com','1-1-2017');

INSERT INTO WORKER

VALUES

(3,'karim','4500','0181xxxxx', 'karim@gmail.com','7-7-2018');

INSERT INTO WORKER

VALUES

(2,'jamal','4000','0181xxxxx', 'jamal@gmail.com','3-7-2017');

INSERT INTO WORKER

VALUES

(4,'Rahim','5000','0181xxxxx', 'Rahim@gmail.com','2-2-2019');

ID	NAME	SALARY	PHONE_NO	EMAIL	HIREDATE
1	kamal	2000	0181xxxxx	kamal@gmaiL.com	1-1-2017
3	karim	4500	0181xxxxx	karim@gmail.com	7-7-2018
2	jamal	4000	0181xxxxx	jamal@gmail.com	3-7-2017
4	Rahim	5000	0181xxxxx	Rahim@gmail.com	2-2-2019

4 rows returned in 0.02 seconds CSV Export

c. STOCK

```
INSERT INTO STOCK
VALUES
('10 TYPES','VERY GOOD','1 HOUSE','21-01-2022');

INSERT INTO STOCK
VALUES
('12 TYPES','GOOD','1 HOUSE','11-01-2022');

INSERT INTO STOCK
VALUES
('12 TYPES','GOOD','1 HOUSE','11-01-2022');

INSERT INTO STOCK
VALUES
('7 TYPES','NORMAL','1 HOUSE','11-04-2022');
```

S_VARIETY	S_QUALITY	S_QUANTITY	S_DATE
10 TYPES	VERY GOOD	1 HOUSE	21-01-2022
12 TYPES	GOOD	1 HOUSE	11-01-2022
12 TYPES	GOOD	1 HOUSE	11-01-2022
7 TYPES	NORMAL	1 HOUSE	11-04-2022

4 rows returned in 0.00 seconds CSV Export

d. FOOD

```
INSERT INTO FOOD
VALUES
(701,'BISCUIT','10000','300');

INSERT INTO FOOD
VALUES
(702,'BREAD','20000','100');

INSERT INTO FOOD
VALUES
(703,'CAKE','30000','150');

INSERT INTO FOOD
VALUES
(704,'CHIPS','50000','80');
```

ID	NAME	QUATITY	PRICE
701	BISCUIT	10000	300
702	BREAD	20000	100
703	CAKE	30000	150
704	CHIPS	50000	80

4 rows returned in 0.00 seconds CSV Export

e. CUSTOMER

```
INSERT INTO CUSTOMER
VALUES
(1,'ABIR','0181xxxxx','KURIL, DHAKA,20/A');

INSERT INTO CUSTOMER
VALUES
(2,'ZAYAD','0181xxxxx','KURIL, DHAKA,10/A');

INSERT INTO CUSTOMER
VALUES
(3,'FAYAZ','0181xxxxx','DHANMONDI, DHAKA,20/C');

INSERT INTO CUSTOMER
VALUES
(4,'SRABONE','0181xxxxx','BASHUNDHARA, DHAKA,2/D');
```

ID	NAME	PHONE_NO	ADDRESS
1	ABIR	0181xxxxx	KURIL,DHAKA,20/A
2	ZAYAD	0181xxxxx	KURIL,DHAKA,10/A
3	FAYAZ	0181xxxxx	DHANMONDI,DHAKA,20/C
4	SRABONE	0181xxxxx	BASHUNDHARA,DHAKA,2/D

4 rows returned in 0.02 seconds CSV Export

f. VISITOR

```
INSERT INTO VISITOR
VALUES
(101,'NAHID','0181xxxxx','GULSHAN, DHAKA,3/D');

INSERT INTO VISITOR
VALUES
(102,'SHAMIM','0181xxxxx','RAMPURA, DHAKA,20/G');

INSERT INTO VISITOR
VALUES
(103,'SHAWN','0181xxxxx','BASHABO, DHAKA,1/5');

INSERT INTO VISITOR
VALUES
(104,'AKRAM','0181xxxxx','NORDA, DHAKA,5/E');
```

ID	NAME	PHONE_NO	ADDRESS
101	NAHID	0181xxxxx	GULSHAN,DHAKA,3/D
102	SHAMIM	0181xxxxx	RAMPURA,DHAKA,20/G
103	SHAWN	0181xxxxx	BASHABO,DHAKA,1/5
104	AKRAM	0181xxxxx	NORDA,DHAKA,5/E

4 rows returned in 0.00 seconds CSV Export

g. MACHINE

```
INSERT INTO MACHINE
VALUES
('999','AGM','2-3-2017');

INSERT INTO MACHINE
VALUES
('909','VSM/F','5-3-2017');

INSERT INTO MACHINE
VALUES
('903','LABMIX','20-7-2018');

INSERT INTO MACHINE
VALUES
('900','MARINATER','6-7-2018');
```

CODE_NO	NAME	PURCHASE_DATE
999	AGM	2-3-2017
909	VSM/F	5-3-2017
903	LABMIX	20-7-2018
900	MARINATER	6-7-2018

4 rows returned in 0.00 seconds CSV Export

h. RECORDS

```
INSERT INTO RECORDS
VALUES
('53 ATTENDS','100 KG','VERY GOOD','52 KG');

INSERT INTO RECORDS
VALUES
('47 ATTENDS','150 KG','VERY GOOD','35 KG');

INSERT INTO RECORDS
VALUES
('57 ATTENDS','78 KG','VERY GOOD','45 KG');

INSERT INTO RECORDS
VALUES
('44 ATTENDS','109 KG','VERY GOOD','78 KG');
```

WORKER_ATTENDANCE	DAILY_SELL	PRODUCT_QUANTITY	REMAINING_STOCKS
53 ATTENDS	100 KG	VERY GOOD	52 KG
47 ATTENDS	150 KG	VERY GOOD	35 KG
57 ATTENDS	78 KG	VERY GOOD	45 KG
44 ATTENDS	109 KG	VERY GOOD	78 KG

4 rows returned in 0.00 seconds CSV Export

9. Query Perform:

- **Using SUBQUERY to show NAME, SALARY, ID of the Workers where their SALARY are greater than '*kamal*'.**

```
SELECT NAME, SALARY, ID
FROM WORKER
WHERE SALARY >
        (SELECT SALARY
        FROM WORKER
        WHERE NAME = 'kamal'
        );
```

NAME	SALARY	ID
karim	4500	3
jamal	4000	2
Rahim	5000	4

3 rows returned in 0.00 seconds CSV Export

- **Using MULTIPLE-ROW SUBQUERY with ANY (It compares value to each value returned by the subquery) operator to show NAME, ID, ADDRESS from VISITOR table where IDs are greater than the ID of '*NAHID*'.**

```
SELECT NAME, ID, ADDRESS
FROM VISITOR
WHERE ID > ANY
        (SELECT ID
        FROM VISITOR
        WHERE NAME = 'NAHID'
        );
```

NAME	ID	ADDRESS
SHAMIM	102	RAMPURA,DHAKA,20/G
SHAWN	103	BASHABO,DHAKA,1/5
AKRAM	104	NORDA,DHAKA,5/E

3 rows returned in 0.00 seconds CSV Export

- **Using Table Alice in table WORKER & CUSTOMER to show the NAME of the workers & customers where their IDs are same in digit.**

```
SELECT WORKER.NAME AS W_NAME, CUSTOMER.NAME AS C_NAME
FROM   WORKER, CUSTOMER
WHERE WORKER.ID = CUSTOMER.ID;
```

W_NAME	C_NAME
kamal	ABIR
karim	FAYAZ
jamal	ZAYAD
Rahim	SRABONE

4 rows returned in 0.00 seconds CSV Export

- **Using SINGLE-ROW SUBQUERY from table WORKER to show NAME, HIREDATE where PHONE_NO is equal to the worker whose ID is 2 and also their(worker) SALARYs are more than his SALARY whose ID is 2**

```
SELECT NAME, HIREDATE
FROM WORKER
WHERE PHONE_NO =
            (SELECT PHONE_NO
            FROM    WORKER
             WHERE   ID = 2)
```

```
AND SALARY >
            (SELECT SALARY
           FROM    WORKER
           WHERE   ID= 2);
```

NAME	HIREDATE
karim	7-7-2018
Rahim	2-2-2019

2 rows returned in 0.00 seconds CSV Export

- **Updating Worker's SALARY using specific condition**

```
UPDATE WORKER
SET SALARY= (SALARY + 2000)
WHERE ID = 4;
```

```
1 row(s) updated.
```

0.00 seconds

- **Creating a view from WORKER table**

```
CREATE VIEW salvu40 AS
SELECT ID WORKER_NUMBER, NAME W_NAME, SALARY W_SALARY
FROM WORKER
WHERE ID = 3;
```

```
View created.
```

0.02 seconds

- **Joining a table ITSELF**

```
SELECT WORKER.NAME || ' makes ' || FOOD.NAME
FROM WORKER, FOOD
WHERE WORKER.SALARY < > FOOD.PRICE;
```

WORKER.NAME||'MAKES'||FOOD.NAME
kamal makes BISCUIT
karim makes BISCUIT
jamal makes BISCUIT
Rahim makes BISCUIT
kamal makes BREAD
karim makes BREAD
jamal makes BREAD
Rahim makes BREAD
kamal makes CAKE
karim makes CAKE
More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.00 seconds CSV Export

- **CHECK Constraint**

```
ALTER TABLE WORKER
ADD AGE SMALLINT
CHECK (AGE>=18);
```

```
Table altered.
```

0.05 seconds

10. Conclusion:

This is a very small scenario of a "Food Factory Management System". The main lacking of this application is its small feature. In real life the management system is very big and huge, and it stores huge amount of data. But it might be helpful for beginners, who want to gain ideas about this management system and want to start a new beginning in this profession.

We took help from internet for ideas about this management. We think, as much as resources we follow, we can gain more knowledge and implement it in our project. As we are the new learners in this database platform, we try our best to make this management system more connected in real life. Hope you will pardon our mistakes.

Resources:

- Slides provided by our honorable faculty.
- https://www.geeksforgeeks.org/normal-forms-in-dbms/#:~:text=Normalization%20is%20the%20process%20of,reduce%20redundancy%20in%20database%20tables.
- https://www.google.com/search?q=normaLIZATION+applied+IN+ER+diagram&tbm=isch&ved=2ahUKEwjg4arDmZ73AhX_k9gFHTgxChEQ2-cCegQIABAA&oq=normaLIZATION+applied+IN+ER+diagram&gs_lcp=CgNpbWcQAzoHCCMQ7wMQJ1DhCFjeIGDBK2gAcAB4AYABmAyIAYY0kgERMC4xLjAuMS4xLjMuMC4yLjGYAQCgAQGqAQtnd3Mtd2l6LWltZ8ABAQ&sclient=img&ei=PahdYqDPN_-n4t4PuOKoiAE&bih=649&biw=1396#imgrc=DkEdeHyDZGsK0M
- https://www.youtube.com/watch?v=g2yF2gyaN7I
- https://tutorialink.com/dbms/normal-forms.dbms

THANK YOU