IE2042 - Database Management Systems for Security

Assignment 01 : 2020 Regular Intake

Title : **Restaurant Management System**

Group Members:

IT Number	Name
IT19029146	Eranda H. P. D (Leader)
IT19040936	S. M. Rathnayaka
IT19038742	J. A. T. N. Weerasooriya
IT19029214	J. C. Hapuarachchi

Contribution for the Project

IT Number	Contribution
IT19029146 (Leader)	1) ER Diagram 2) Relational Schema 3) **Creating tables and inserting data :-** • Payment • Employee • Cart • Delivery_Person • Customer_C_Email • Chef_Food_Item 4) Implementing Access Control Privileges for the database 5) Implementing countermeasure methods for the database (View, Stored Procedure, Transparent Data Encryption)
IT19040936	1) **Creating tables and inserting data :-** • Orders • Chef • Delivery • Receptionist • Customer_Phone_number 2) Completed 2 transaction operations. 3) Explained 3 web based database attack types and countermeasures for them. (Database Backups Exposure, DoS Attack, Exploitation of Vulnerable Misconfigured Databases)
IT19038742	1) **Creating tables and inserting data :-** • User_account • Delivery_Food_Item • Manager • Waiter • Employee_Phone 2) Explained 3 web based database attack types and countermeasures for them. (Excessive Database privileges, Database vulnerabilities and misconfigurations, Limited Security Expertise and Education) 3) Explained database errors and recovery mechanisms

IT19029214	1) **<u>Creating tables and inserting data :-</u>** • Customer • Food_Item • DB_Admin • Employee_Email • Waiter_Food_Item 2) Completed 2 transaction operations. 3) Explained 4 web based database attack types and countermeasures for them. (SQL Injection, Weak Audit Trail, Unmanaged sensitive data, Inadequate Database Backups)

Table of Contents

1. ER Diagram 5

2. Relational Schema 7

3. Table Creation Queries 10

4. All the tables with data 12

5. Transaction Operations 21

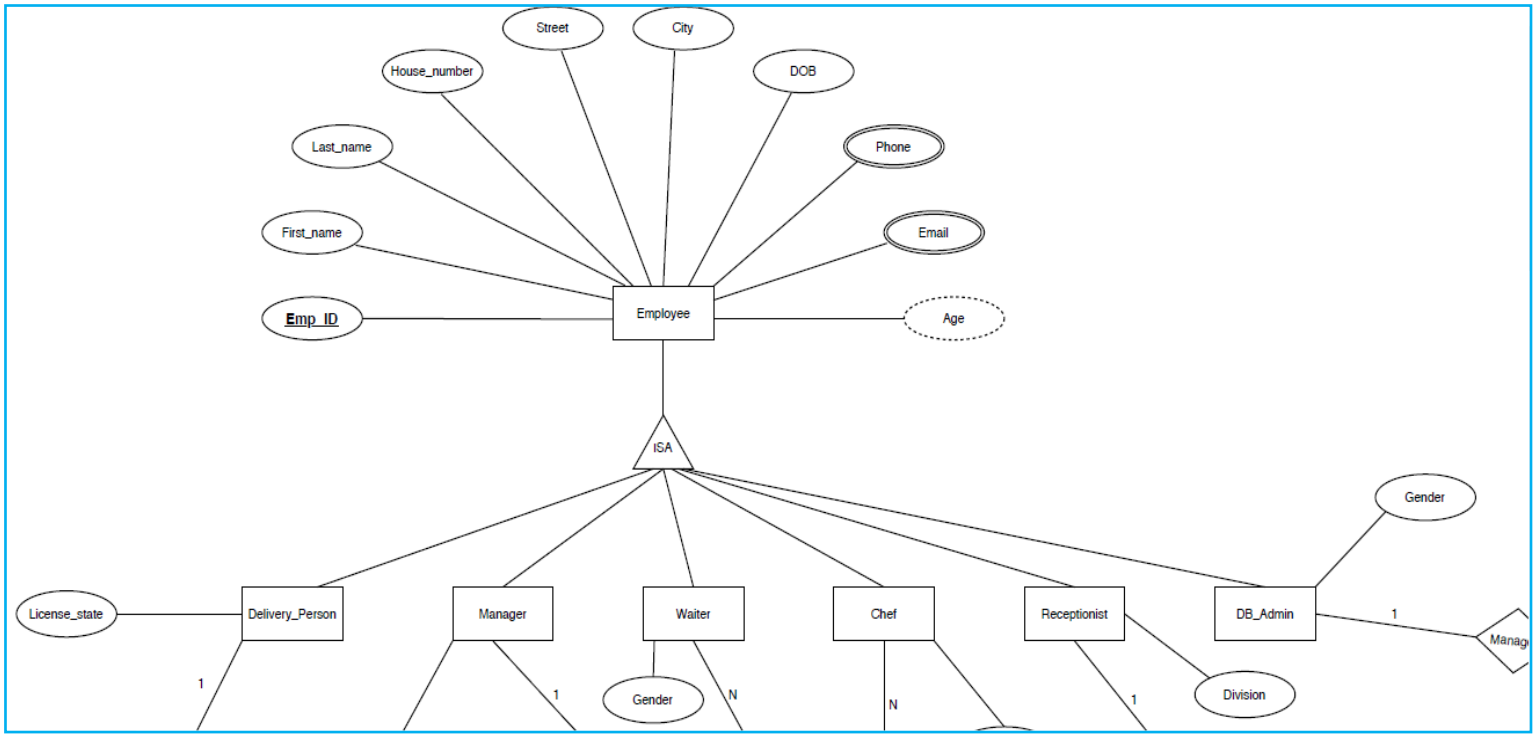
6. Implementation of Access Control Privileges 23

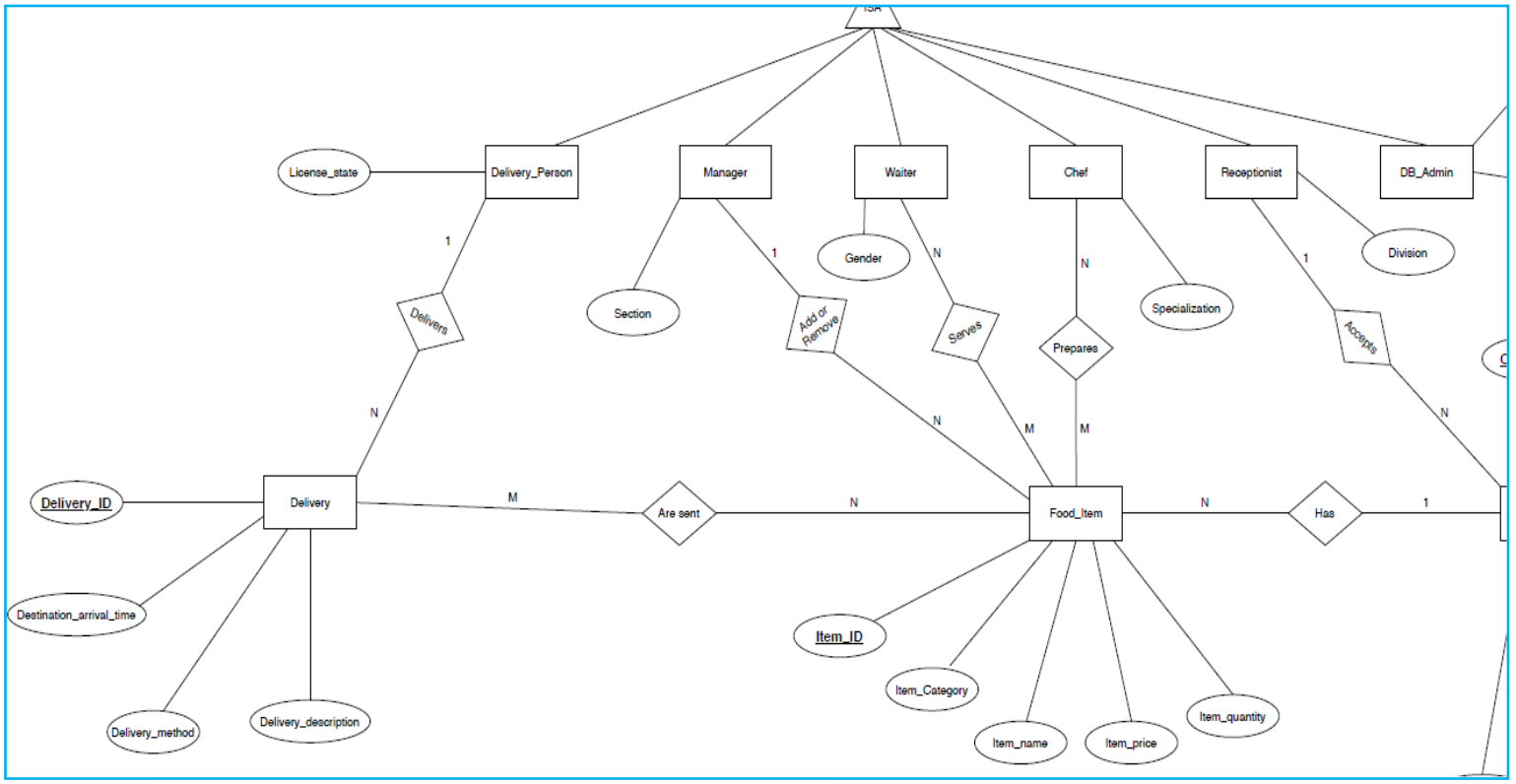
7. Attacks and Countermeasures (Web-Based database) 26

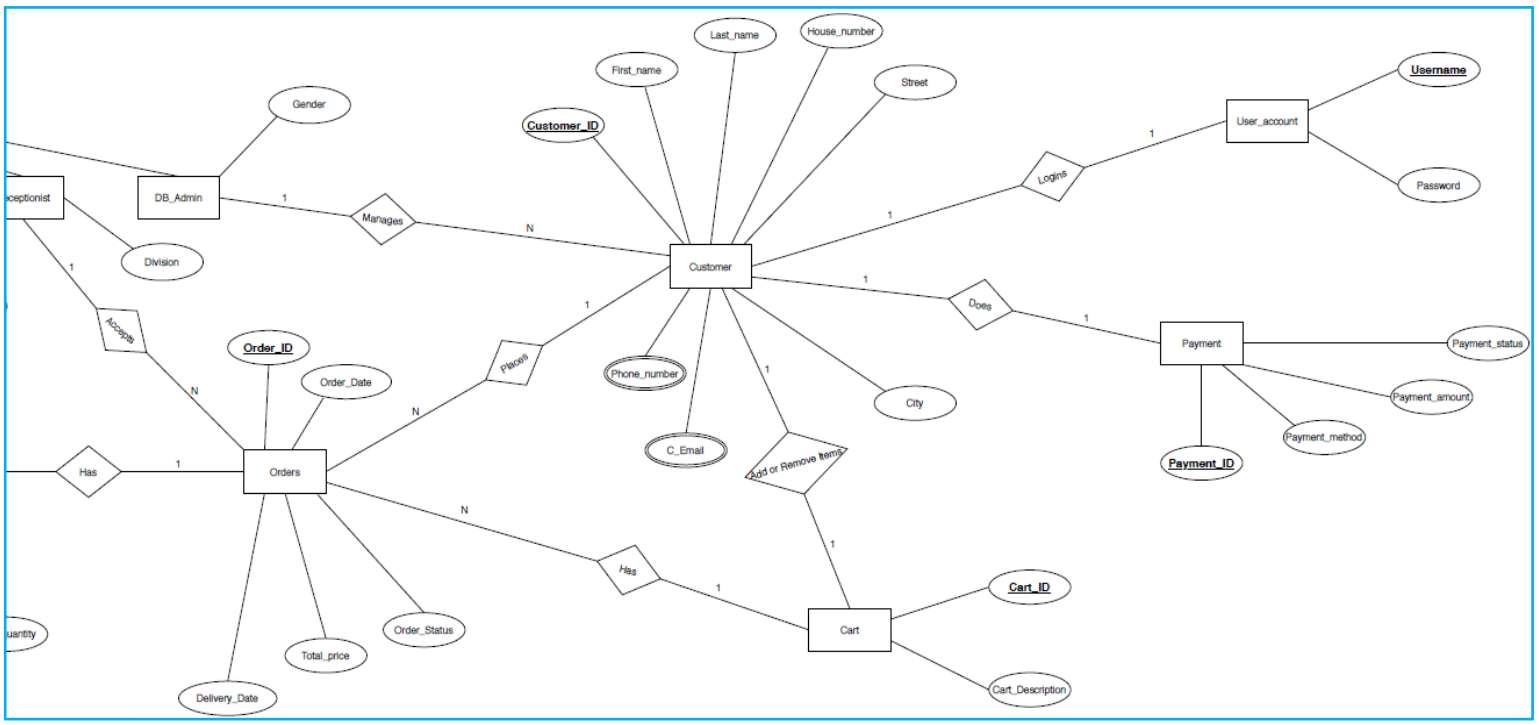
8. Implementation of Additional Security Mechanisms 42

9. Database Recovery Mechanisms 46

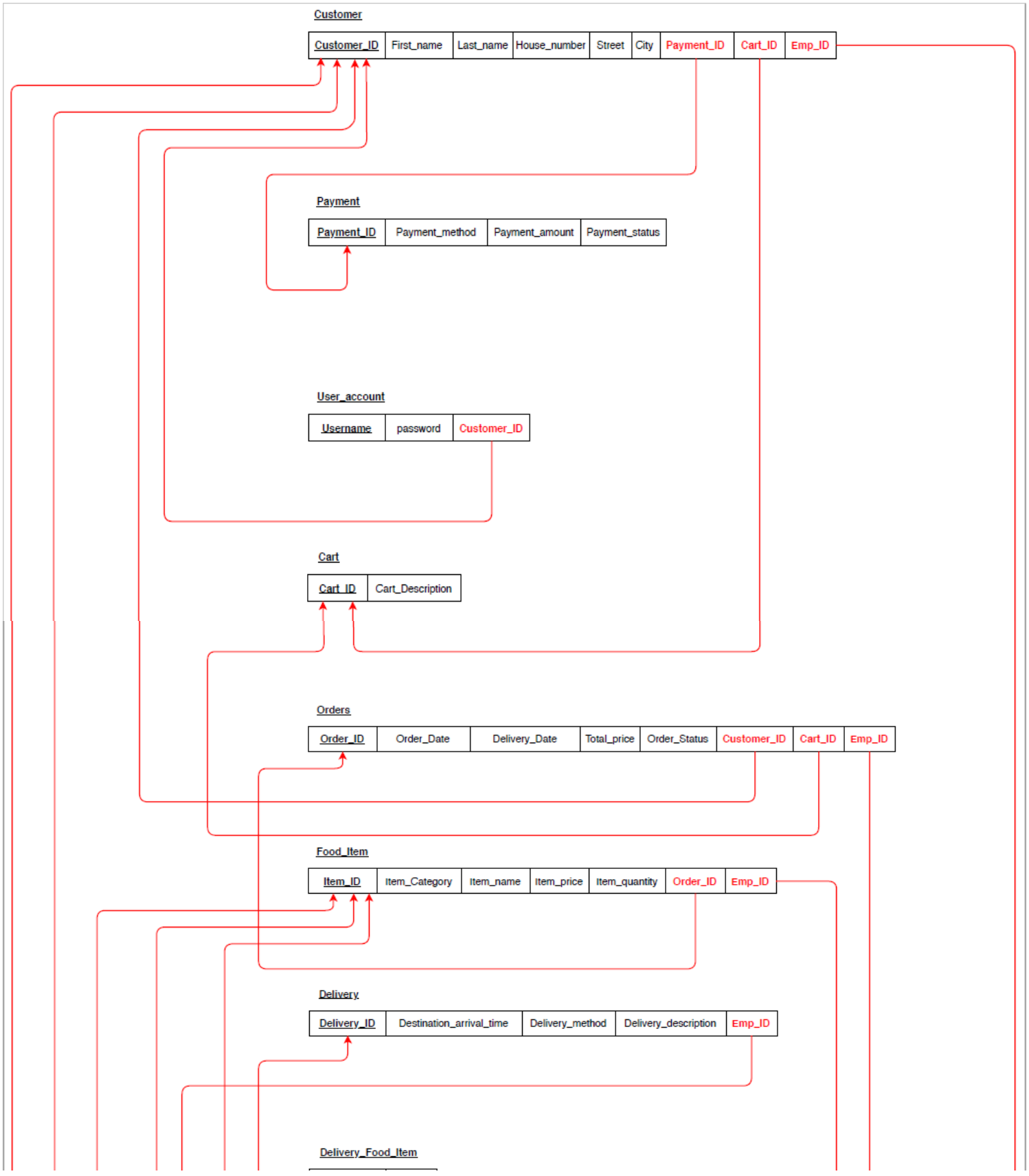
ER Diagram

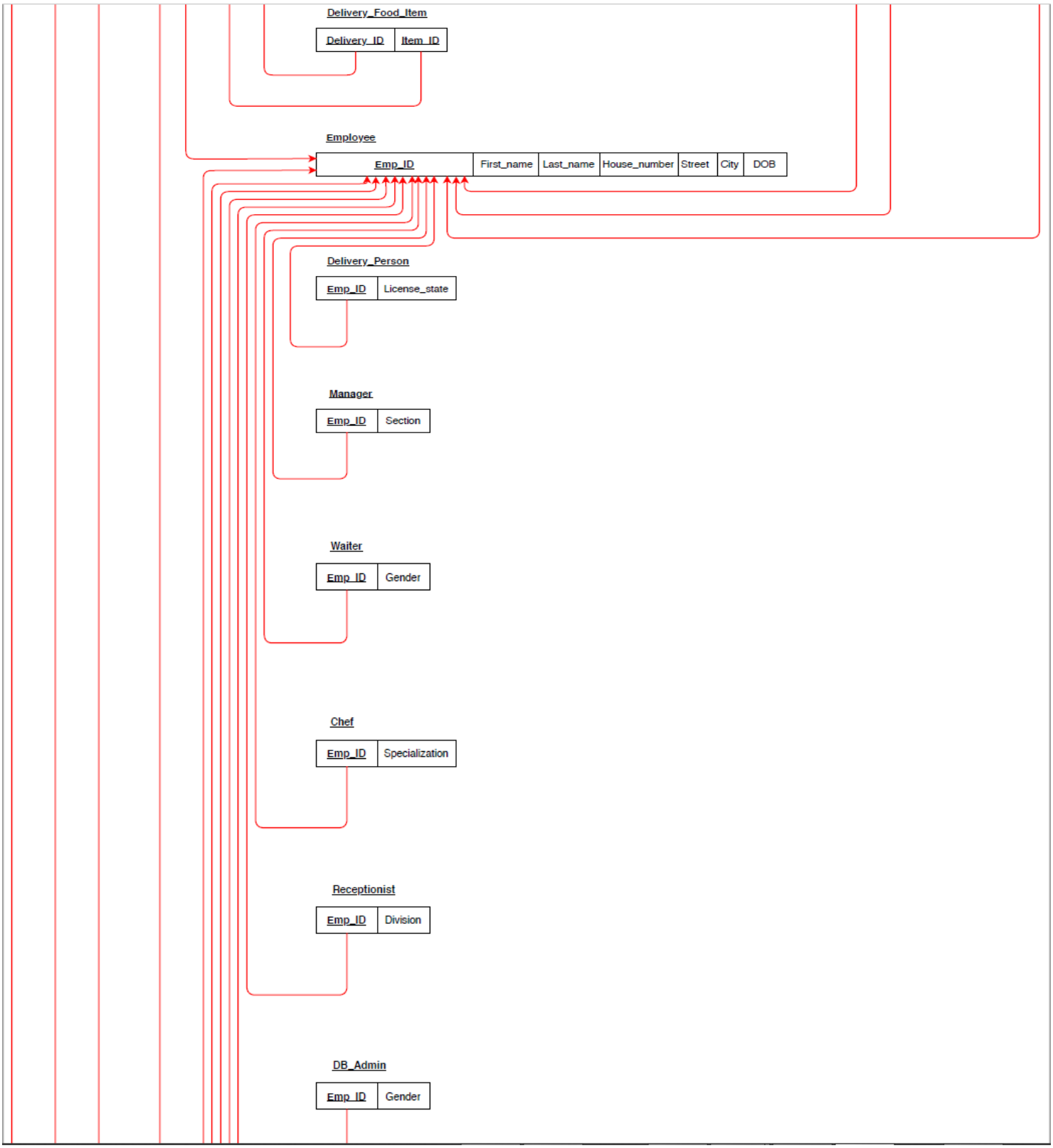

Street
City
House_number
DOB
Last_name
Phone
First_name
Email
Emp_ID
Employee
Age
ISA
Gender
License_state
Delivery_Person
Manager
Waiter
Chef
Receptionist
DB_Admin
1
Division
Gender
N
1
1
N
1


License_state
Delivery_Person
Manager
Waiter
Chef
Receptionist
DB_Admin
Division
Gender
Section
Specialization
Delivers
Add or Remove
Serves
Prepares
Accepts
1
N
M
Delivery_ID
Delivery
Are sent
Food_Item
Has
Destination_arrival_time
Item_ID
Item_Category
Item_name
Item_price
Item_quantity
Delivery_method
Delivery_description


Last_name
House_number
First_name
Street
Gender
Customer_ID
Username
User_account
DB_Admin
Logins
Password
Division
Manages
Customer
Does
Payment
Payment_status
Accepts
Order_ID
Order_Date
Places
Phone_number
City
Payment_amount
Payment_method
C_Email
Payment_ID
Has
Orders
Add or Remove Items
Has
Cart_ID
Cart
Order_Status
Total_price
Delivery_Date
Cart_Description

Relational Schema




Delivery_Food_Item
Delivery_ID
Item_ID
Employee
Emp_ID
First_name
Last_name
House_number
Street
City
DOB
Delivery_Person
Emp_ID
License_state
Manager
Emp_ID
Section
Waiter
Emp_ID
Gender
Chef
Emp_ID
Specialization
Receptionist
Emp_ID
Division
DB_Admin
Emp_ID
Gender

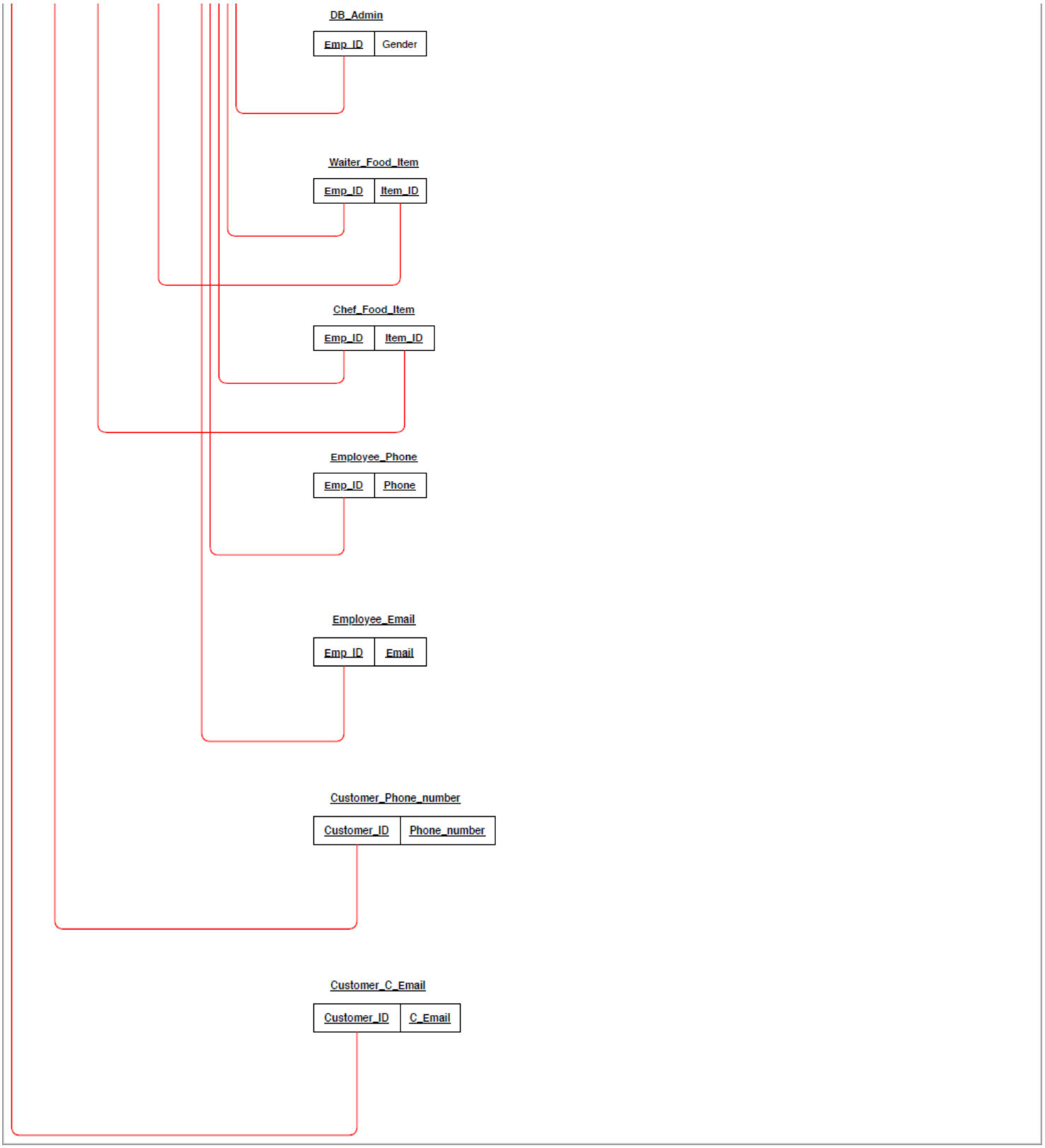

DB_Admin
Emp_ID
Gender
Waiter_Food_Item
Emp_ID
Item_ID
Chef_Food_Item
Emp_ID
Item_ID
Employee_Phone
Emp_ID
Phone
Employee_Email
Emp_ID
Email
Customer_Phone_number
Customer_ID
Phone_number
Customer_C_Email
Customer_ID
C_Email

Table Creation Queries

```
--Employee Table--

CREATE TABLE Employee(

                Emp_ID VARCHAR(20) NOT NULL,
                First_name VARCHAR(50),
                Last_name VARCHAR(50),
                House_number VARCHAR(50),
                Street VARCHAR(40),
                City VARCHAR(20),
                DOB DATE,

                CONSTRAINT Employee_PK PRIMARY KEY (Emp_ID)

                    );


--Payment Table--

CREATE TABLE Payment(

                Payment_ID VARCHAR(20) NOT NULL,
                Payment_method VARCHAR(50),
                Payment_amount REAL,
                Payment_status VARCHAR(40),

                CONSTRAINT Payment_PK PRIMARY KEY (Payment_ID)

                    );
```

```
--Cart Table --

CREATE TABLE Cart(

                Cart_ID VARCHAR(20) NOT NULL,
                Cart_Description VARCHAR(100),

                CONSTRAINT Cart_PK PRIMARY KEY (Cart_ID)

                );


--Customer Table--

CREATE TABLE Customer(
Customer_ID VARCHAR(20) NOT NULL,
First_name VARCHAR(100),
Last_name VARCHAR(100),
House_number char(10),
Street VARCHAR(50),
City VARCHAR(50),
Payment_ID VARCHAR(20),
Cart_ID VARCHAR(20),
Emp_ID VARCHAR(20),
CONSTRAINT pk_Customer_ID PRIMARY KEY (Customer_ID),
CONSTRAINT fk1_Customer FOREIGN KEY (Payment_ID) REFERENCES Payment (Payment_ID),
CONSTRAINT fk2_Customer FOREIGN KEY (Cart_ID) REFERENCES Cart (Cart_ID),
CONSTRAINT fk3_Customer FOREIGN KEY (Emp_ID) REFERENCES Employee (Emp_ID)
);
```

```sql
-- Orders Table--

CREATE TABLE Orders(

Order_ID  VARCHAR(20) NOT NULL,
Order_Date DATE,
Delivery_Date DATE,
Total_price REAL,
Order_Status VARCHAR(20),
Customer_ID VARCHAR(20),
Cart_ID  VARCHAR(20),
Emp_ID VARCHAR(20),

CONSTRAINT pk_order PRIMARY KEY (Order_ID),
CONSTRAINT fr_order FOREIGN KEY (Customer_ID) REFERENCES Customer(Customer_ID),
CONSTRAINT fr_order2 FOREIGN KEY (Emp_ID) REFERENCES Employee(Emp_ID),
CONSTRAINT fr_orrder3 FOREIGN KEY (Cart_ID) REFERENCES Cart(Cart_ID)

);


-- Food_Item Table--

CREATE TABLE Food_Item
(
Item_ID VARCHAR(20) NOT NULL,
Item_Category VARCHAR(50),
Item_name VARCHAR (100),
Item_price real,
```

All the Tables with Data

1) Employee Table

	EMP_ID	FIRST_NAME	LAST_NAME	HOUSE_NUMBER	STREET	CITY	DOB
1	M1234	Amal	Perera	10/A	1st Lane	Peradeniya	11-JAN-75
2	M5678	Kamal	Silva	12/B	2nd Lane	Kandy	05-APR-78
3	M1357	Nimal	Kumara	15/A	3rd Lane	Gampaha	20-JUN-76
4	M2468	Sunil	Santha	13/C	1st Lane	Colombo	22-AUG-69
5	M1020	Anil	Jayasinghe	14/B	2nd Lane	Kaluthara	14-MAR-62
6	M3123	Amarabandu	Rupasinghe	3/B	1st Lane	Galle	23-MAY-64
7	M2456	Sirisena	Herath	11/A	3rd Lane	Polonnaruwa	28-FEB-79
8	M3463	Tharaka	Gamlath	17/B	2nd Lane	Rathnapura	08-AUG-88
9	M4267	Kumar	Dharmasena	13/A	1st Lane	Hambanthota	05-MAY-75
10	M1024	Amarasiri	Fernando	16/C	3rd Lane	Kandy	07-JUL-87
11	M3483	Tharaka	Silva	18/A	1st Lane	Gampaha	11-JAN-71
12	M5648	Thushal	Kulathilaka	11/B	2nd Lane	Kandy	05-APR-72
13	M1050	Eishan	Dinuka	14/A	3rd Lane	Gampaha	20-JUN-73
14	M1950	Amara	Perera	15/C	5th Lane	Colombo	22-AUG-79
15	M1750	Kasun	Kalhara	16/B	2nd Lane	Kaluthara	14-MAR-73
16	M10	Yasas	Chandrasiri	7/B	4th Lane	Galle	23-MAY-74
17	M20	Dilakshi	Sandumini	3/A	3rd Lane	Polonnaruwa	28-FEB-75
18	M2122	Kavisha	theekshana	5/B	7th Lane	Rathnapura	08-AUG-81
19	M2322	Kavindu	Lakshan	7/A	1st Lane	Hambanthota	05-MAY-82
20	M2422	Srimal	Perera	8/C	3rd Lane	Kandy	07-JUL-82
21	M2522	Sahan	Kavinga	9/A	1st Lane	Gampaha	11-JAN-83
22	M2622	Thisal	Pulsitha	10/B	4th Lane	Kandy	05-APR-84
23	M1211	Inura	Kumara	1/A	3rd Lane	Nuwaraeliya	20-JUN-85
24	M1311	Janith	Sankalpa	13/C	6th Lane	Colombo	22-AUG-86
25	M1411	Sandul	Lakshan	2/B	8th Lane	Mathara	14-MAR-87
26	M1511	Sanuka	Wickramasinghe	4/B	1st Lane	Galle	23-MAY-88
27	M1611	Pasan	Herath	3/A	3rd Lane	Polonnaruwa	28-FEB-89
28	M1235	Tirone	Silva	5/B	2nd Lane	Rathnapura	08-AUG-90
29	M1358	Avishka	Fernando	5/A	1st Lane	Hambanthota	05-MAY-91
30	M2459	Kusal	Mendis	8/C	3rd Lane	Mathara	07-JUL-92
31	M3473	Nuwan	Madushan	7/A	1st Lane	Gampaha	11-JAN-93
32	M1078	Hashan	Perera	13/B	2nd Lane	Kandy	05-APR-94
33	M4253	Shamal	Fonseka	15/A	6th Lane	Gampaha	20-JUN-92
34	M1111	Kavinga	Munidasa	13/A	1st Lane	Colombo	22-AUG-91
35	M2222	Lahiru	Madushanka	14/B	2nd Lane	Mathara	14-MAR-90
36	M3333	Bhanuka	Rajapaksha	3/B	1st Lane	Galle	23-MAY-89
37	M4444	Nawodya	Shehan	11/C	10th Lane	Polonnaruwa	28-FEB-88
38	M5555	Pasindu	Gamlath	17/D	2nd Lane	Mathara	08-AUG-87
39	M1334	Kavinga	Herath	12/A	1st Lane	Colombo	22-AUG-92
40	M1434	Lahiru	Ambegoda	13/B	2nd Lane	Ampara	14-MAR-94
41	M1534	Bhanuka	Udawatte	12/B	1st Lane	Kelaniya	23-MAY-85
42	M1634	Nawodya	Soysa	12/C	10th Lane	Rathnapura	28-FEB-83

2) Payment Table

	PAYMENT_ID	PAYMENT_METHOD	PAYMENT_AMOUNT	PAYMENT_STATUS
1	PN1010	Master Card	6000	Successful
2	PN2020	Visa Card	3000	Successful
3	PN3030	Cash On Delivery	6200	Pending
4	PN4040	Master Card	2500	Successful
5	PN5050	Cash	1200	Successful

3) Cart Table

	CART_ID	CART_DESCRIPTION
1	C10	Description_1
2	C20	Description_2
3	C30	Description_3
4	C40	Description_4
5	C50	Description_5

4) Customer Table

	CUSTOMER_ID	FIRST_NAME	LAST_NAME	HOUSE_NUMBER	STREET	CITY	PAYMENT_ID	CART_ID	EMP_ID
1	CUS001	Shalini	Fernando	No.40	Jayasiri Rd	Nattandiya	PN1010	C10	M1111
2	CUS002	Gehan	Tharaka	No.50/B	Udawala Rd	Hambanthota	PN2020	C20	M2222
3	CUS003	Shehani	Perera	No.11/A	Welivita Rd	Malabe	PN3030	C30	M3333
4	CUS004	Jayani	Shahini	No.2/C2	Devala Rd	Wennappuwa	PN4040	C40	M4444
5	CUS005	Wiajaya	Silva	No.20	Nivasa Rd	Pitakotuwa	PN5050	C50	M5555

5) User_account Table

	USERNAME	PASSWORD	CUSTOMER_ID
1	jayani	jayani123	CUS001
2	samadhi	samadhi123	CUS002
3	eranda	eranda123	CUS003
4	tharindu	tharindu123	CUS004
5	chamodhi	chamodhi123	CUS005

6) Orders Table

	ORDER_ID	ORDER_DATE	DELIVERY_DATE	TOTAL_PRICE	ORDER_STATUS	CUSTOMER_ID	CART_ID	EMP_ID
1	001	23-SEP-19	23-AUG-19	500	complete	CUS001	C10	M1234
2	002	23-AUG-19	23-SEP-19	1000	complete	CUS002	C20	M1334
3	003	23-AUG-19	23-OCT-19	550	Notcomplete	CUS003	C30	M1434
4	004	23-OCT-19	23-SEP-19	1500	Complete	CUS004	C40	M1534
5	005	23-AUG-19	23-AUG-19	5500	complete	CUS005	C50	M1634

7) Food_Item Table

	ITEM_ID	ITEM_CATEGORY	ITEM_NAME	ITEM_PRICE	ITEM_QUANTITY	ORDER_ID	EMP_ID
1	FID001	Bakery	Fish Bun	70	40	001	M1111
2	FID002	Spicy	Rolls	50	30	002	M2222
3	FID003	Bakery	Sugar Bun	25	40	003	M3333
4	FID004	Spicy	Patis	40	30	004	M4444
5	FID005	Bakery	Pastry	50	40	005	M5555

8) Delivery Table

	DELIVERY_ID	DESTINATION_ARRIVAL_TIME	DELIVERY_METHOD	DELIVERY_DESCRIPTION	EMP_ID
1	D001	19-AUG-23	Mini	Description1	M1234
2	D002	23-SEP-19	Mass	Description2	M1334
3	D003	23-OCT-19	Mini	Description3	M1434
4	D004	23-SEP-19	Mass	Description4	M1534
5	D005	23-AUG-19	Mini	Description5	M1634

9) Delivery_Food_Item Table

	DELIVERY_ID	ITEM_ID
1	D001	FID001
2	D002	FID002
3	D003	FID003
4	D004	FID004
5	D005	FID005

10) Delivery_Person Table

	EMP_ID	LICENSE_STATE
1	M1357	Valid
2	M3123	Expired
3	M2456	Valid
4	M3463	Valid
5	M4267	Valid

11) Manager Table

	EMP_ID	SECTION
1	M2122	Delivery
2	M2322	Rice
3	M2422	Dessert
4	M2522	Drinks
5	M2622	Delivery

12) Waiter Table

	EMP_ID	GENDER
1	M1211	Male
2	M1311	Female
3	M1411	Female
4	M1511	Male
5	M1611	Male

13) Chef Table

	EMP_ID	SPECIALIZATION
1	M1235	Western foods
2	M1358	Indian foods
3	M2459	Chinese foods
4	M3473	Sri lankan foods
5	M1078	Indian foods
6	M4253	Western foods

14) Receptionist Table

	EMP_ID	DIVISION
1	M3483	Ordering Division
2	M5648	Inquiry Division
3	M1050	Financial Division
4	M1950	Ordering Division
5	M1750	Inquiry Division

15) DB_Admin Table

	EMP_ID	GENDER
1	M10	Male
2	M20	Female

16) Waiter_Food_Item Table

	EMP_ID	ITEM_ID
1	M1211	FID001
2	M1311	FID002
3	M1411	FID003
4	M1511	FID004
5	M1611	FID005

17) Chef_Food_Item Table

	EMP_ID	ITEM_ID
1	M1078	FID005
2	M1235	FID001
3	M1358	FID002
4	M2459	FID003
5	M3473	FID004

18) Customer_Phone_number Table

	CUSTOMER_ID	PHONE_NUMBER
1	CUS001	0750587963
2	CUS002	0769875321
3	CUS003	0705878792
4	CUS004	078007963
5	CUS005	0760796345

19) Customer_C_Email Table

	CUSTOMER_ID	C_EMAIL
1	CUS001	chamodhij@gmail.com
2	CUS002	dilshane@gmail.com
3	CUS003	samadhim@gmail.com
4	CUS004	tharindux@gmail.com
5	CUS005	amarabandur@gmail.com

20) Employee_Phone Table

	EMP_ID	PHONE
1	M1050	0775623442
2	M1078	0778098675
3	M1111	0776756453
4	M1211	0776786653
5	M1234	0778774275
6	M1235	0776234553
7	M1311	0712347836
8	M1334	0776237553
9	M1358	0712764786
10	M1411	0785634653
11	M1434	0712758786
12	M1511	0775623442
13	M1534	0785876453
14	M1611	0778763275
15	M1634	0778678742
16	M1750	0778776275
17	M2122	0773456453
18	M2222	0712234786
19	M2322	0716734786
20	M2422	0789076453
21	M2456	0712382836
22	M2459	0785656453
23	M2522	0775623342
24	M2622	0778746675
25	M3123	0776784653
26	M3333	0785676453
27	M3463	0785694653
28	M3473	0778675342
29	M3483	0715677836
30	M4253	0776324653
31	M4267	0775661442
32	M4444	0775645342
33	M5555	0778798675
34	M5648	0785864653

21) Employee_Email Table

	EMP_ID	EMAIL
1	M1078	bb2@gmail.com
2	M1111	jay@gmail.com
3	M1234	ttt2@gmail.com
4	M1235	jay@gmail.com
5	M1311	ggg2@gmail.com
6	M1334	uuu2@gmail.com
7	M1358	emp12@gmail.com
8	M1411	hhh2@gmail.com
9	M1434	www2@gmail.com
10	M1511	iiii2@gmail.com
11	M1534	xxx2@gmail.com
12	M1611	jjj2@gmail.com
13	M1634	yyy2@gmail.com
14	M1750	fff2@gmail.com
15	M1950	eee2@gmail.com
16	M2122	kkk2@gmail.com
17	M2222	emp12@gmail.com
18	M2322	lll2@gmail.com
19	M2422	mmm2@gmail.com
20	M2456	qqq2@gmail.com
21	M2459	sen123@gmail.com
22	M2522	nnn2@gmail.com
23	M2622	ooo2@gmail.com
24	M3123	ppp2@gmail.com
25	M3333	sen123@gmail.com
26	M3463	rrr2@gmail.com
27	M3473	aaa2@gmail.com
28	M3483	ddd2@gmail.com
29	M4253	ccc2@gmail.com
30	M4267	sss2@gmail.com
31	M4444	yash12@gmail.com
32	M5555	myac@gmail.com
33	M5648	eee3@gmail.com

Transaction Operations

1) Transaction 1

```
UPDATE Food_Item;

SET Item_price= Item_price  +  200;

WHERE Item_ID = ‘F1D002’;


UPDATE Orders;

SET Total_price = Total _ Price + 200;

WHERE Order_ID = ‘002’;

INSERT INTO Payment(Payment_ID, Payment_method, Payment_amount,
Payment_status) VALUES ('PN3030', 'Cash On Delivery', totalpriceinorder, 'Pending');


COMMIT;
```

2) Transaction 2

```
UPDATE Food_Item;

SET Item_price= Item_price  -  150;

WHERE Item_ID = ‘F1D002’;


UPDATE Orders;

SET Total_price = Total _ Price - 150;

WHERE Order_ID = ‘002’;

INSERT INTO Payment(Payment_ID, Payment_method, Payment_amount,
Payment_status) VALUES ('PN3030', 'Cash On Delivery', totalpriceinorder, 'Pending');


COMMIT;
```

3) Transaction 3

```
SET TRANSCTION READ WRITE;
UPDATE Payment
SET Payment_amount = Payment_amount - 250;
WHERE Payment_ID = 'PN1010'

SET Payment_amount = Payment_amount + 250;
WHERE Payment_ID = 'PN2020'

COMMIT;
```

4) Transaction 4

```
SET TRANSCTION READ WRITE;
UPDATE Orders
SET Total_price = Total_price - 500;
WHERE Order_ID = '001'

SET Order_Status = 'not complete';
WHERE Order_ID = '001'

COMMIT;
```

Implementing Access Control Privileges

```
-- Enabling for scripting--
ALTER SESSION SET"_ORACLE_SCRIPT"=TRUE;


-- Creating roles--
CREATE ROLE Customer;
CREATE ROLE Delivery_Person;
CREATE ROLE Manager;
CREATE ROLE Waiter;
CREATE ROLE Chef;
CREATE ROLE Receptionist;
CREATE ROLE DB_Admin;


-- Creating users--
-- (Customers)--
CREATE USER Shalini IDENTIFIED BY ShaliniPW;
CREATE USER Gehan IDENTIFIED BY GehanPW;
CREATE USER Shehani IDENTIFIED BY ShehaniPW;
CREATE USER Jayani IDENTIFIED BY JayaniPW;
CREATE USER Wiajaya IDENTIFIED BY WiajayaPW;

-- (DB_Admin)--
CREATE USER Yasas IDENTIFIED BY YasasPW;
CREATE USER Dilakshi IDENTIFIED BY DilakshiPW;
```

```
-- (Employees)--
CREATE USER Amal IDENTIFIED BY AmalPW;
CREATE USER Kamal IDENTIFIED BY KamalPW;
CREATE USER Nimal IDENTIFIED BY NimalPW;
CREATE USER Sunil IDENTIFIED BY SunilPW;
CREATE USER Anil IDENTIFIED BY AnilPW;
CREATE USER Amarabandu IDENTIFIED BY AmarabanduPW;
CREATE USER Sirisena IDENTIFIED BY SirisenaPW;
CREATE USER Tharaka IDENTIFIED BY TharakaPW;
CREATE USER Kumar IDENTIFIED BY KumarPW;
CREATE USER Amarasiri IDENTIFIED BY AmarasiriPW;
CREATE USER Thushal IDENTIFIED BY ThushalPW;
CREATE USER Eishan IDENTIFIED BY EishanPW;
CREATE USER Amara IDENTIFIED BY AmaraPW;
CREATE USER Kasun IDENTIFIED BY KasunPW;
CREATE USER Kavisha IDENTIFIED BY KavishaPW;
CREATE USER Kavindu IDENTIFIED BY KavinduPW;
CREATE USER Srimal IDENTIFIED BY SrimalPW;
CREATE USER Sahan IDENTIFIED BY SahanPW;
CREATE USER Thisal IDENTIFIED BY ThisalPW;
CREATE USER Inura IDENTIFIED BY InuraPW;
CREATE USER Janith IDENTIFIED BY JanithPW;
CREATE USER Sandul IDENTIFIED BY SandulPW;
CREATE USER Sanuka IDENTIFIED BY SanukaPW;
CREATE USER Pasan IDENTIFIED BY PasanPW;
CREATE USER Tirone IDENTIFIED BY TironePW;
CREATE USER Avishka IDENTIFIED BY AvishkaPW;
CREATE USER Kusal IDENTIFIED BY KusalPW;
CREATE USER Nuwan IDENTIFIED BY NuwanPW;
CREATE USER Hashan IDENTIFIED BY HashanPW;
CREATE USER Shamal IDENTIFIED BY ShamalPW;
CREATE USER Kavinga IDENTIFIED BY KavingaPW;
CREATE USER Lahiru IDENTIFIED BY LahiruPW;
CREATE USER Bhanuka IDENTIFIED BY BhanukaPW;
CREATE USER Nawodya IDENTIFIED BY NawodyaPW;
CREATE USER Pasindu IDENTIFIED BY PasinduPW;
```

```
-- Granting Access Privileges to Roles--
GRANT ALL PRIVILEGES TO DB_Admin;

-- Here a customer should not be able to view other customers' order details.--

GRANT CONNECT TO Customer;           -- allow user to log in to the database--
GRANT insert, update ON Customer TO Customer;
GRANT insert, update ON Payment TO Customer;
GRANT insert, update ON User_account TO Customer;
GRANT insert, update ON Cart TO Customer;
GRANT insert, update ON Orders TO Customer;
GRANT select ON Food_Item TO Customer;
GRANT insert, update ON Delivery TO Customer;
GRANT insert, update ON Customer_Phone_number TO Customer;
GRANT insert, update ON Customer_C_Email TO Customer;

GRANT CONNECT TO Delivery_Person;
GRANT select ON Customer TO Delivery_Person;
GRANT select ON Payment TO Delivery_Person;
GRANT select ON Orders TO Delivery_Person;
GRANT select ON Food_Item TO Delivery_Person;
GRANT select ON Delivery TO Delivery_Person;
GRANT select ON Delivery_Person TO Delivery_Person;
GRANT select ON Customer_Phone_number TO Delivery_Person;
GRANT select ON Employee_Phone TO Delivery_Person;
```

```
GRANT CONNECT TO Waiter;
GRANT select ON Customer TO Waiter;
GRANT select ON Payment TO Waiter;
GRANT select ON Orders TO Waiter;
GRANT select ON Food_Item TO Waiter;
GRANT select ON Waiter TO Waiter;
GRANT select ON Employee_Phone TO Waiter;

GRANT CONNECT TO Chef;
GRANT select ON Customer TO Chef;
GRANT select ON Orders TO Chef;
GRANT select ON Cart TO Chef;
GRANT select ON Chef TO Chef;
GRANT select ON Employee_Phone TO Chef;
```

```
GRANT CONNECT TO Receptionist;
GRANT select ON Customer TO Receptionist;
GRANT select ON Payment TO Receptionist;
GRANT select ON Cart TO Receptionist;
GRANT select ON Orders TO Receptionist;
GRANT select ON Food_Item TO Receptionist;
GRANT select ON Delivery TO Receptionist;
GRANT select ON Employee TO Receptionist;
GRANT select ON Delivery_Person TO Receptionist;
GRANT select ON Manager TO Receptionist;
GRANT select ON Waiter TO Receptionist;
GRANT select ON Chef TO Receptionist;
GRANT select ON Receptionist TO Receptionist;
GRANT select ON DB_Admin TO Receptionist;
GRANT select ON Employee_Phone TO Receptionist;
GRANT select ON Employee_Email TO Receptionist;
GRANT select ON Customer_Phone_number TO Receptionist;
GRANT select ON Customer_C_Email TO Receptionist;
```

```
GRANT CONNECT TO Manager;
GRANT select, insert, update, delete ON Customer TO Manager;
GRANT select, insert, update, delete ON Payment TO Manager;
GRANT select, insert, update, delete ON Cart TO Manager;
GRANT select, insert, update, delete ON Orders TO Manager;
GRANT select, insert, update, delete ON Food_Item TO Manager;
GRANT select, insert, update, delete ON Delivery TO Manager;
GRANT select, insert, update, delete ON Employee TO Manager;
GRANT select, insert, update, delete ON Delivery_Person TO Manager;
GRANT select, insert, update, delete ON Manager TO Manager;
GRANT select, insert, update, delete ON Waiter TO Manager;
GRANT select, insert, update, delete ON Chef TO Manager;
GRANT select, insert, update, delete ON Receptionist TO Manager;
GRANT select, insert, update, delete ON DB_Admin TO Manager;
GRANT select, insert, update, delete ON Employee_Phone TO Manager;
GRANT select, insert, update, delete ON Employee_Email TO Manager;
GRANT select, insert, update, delete ON Customer_Phone_number TO Manager;
GRANT select, insert, update, delete ON Customer_C_Email TO Manager;


-- Granting the users for roles--
GRANT Customer TO Shalini;
GRANT Customer TO Gehan;
GRANT Customer TO Shehani;
GRANT Customer TO Jayani;
GRANT Customer TO Wiajaya;
```

```
GRANT DB_Admin TO Yasas;
GRANT DB_Admin TO Dilakshi;

GRANT Manager TO Kavisha;
GRANT Manager TO Kavindu;
GRANT Manager TO Srimal;
GRANT Manager TO Sahan;
GRANT Manager TO Thisal;

GRANT Delivery_Person TO Nimal;
GRANT Delivery_Person TO Amarabandu;
GRANT Delivery_Person TO Sirisena;
GRANT Delivery_Person TO Tharaka;
GRANT Delivery_Person TO Kumar;

GRANT Waiter TO Inura;
GRANT Waiter TO Janith;
GRANT Waiter TO Sandul;
GRANT Waiter TO Sanuka;
GRANT Waiter TO Pasan;

GRANT Chef TO Tirone;
GRANT Chef TO Avishka;
GRANT Chef TO Kusal;
GRANT Chef TO Nuwan;
GRANT Chef TO Hashan;
GRANT Chef TO Shamal;
```

```
GRANT Receptionist TO Thushal;
GRANT Receptionist TO Eishan;
GRANT Receptionist TO Amara;
GRANT Receptionist TO Kasun;
```

Attacks and Countermeasures of a web-based Database

- Today we all are living in a data centric world. In modern era “data” is the most important and the most valuable asset to any organization. Over the last several years data breaching has risen up by a considerable amount. This has caused massive data losses for both small businesses and world famous business brands.

  - Some of them are as following ;

  1) AOL
  2) MySpace
  3) AT&T
  4) Apple
  5) LinkedIn

- Most of the time databases may contain very sensitive data such as credit card information, personal information etc. So when it comes to databases which are hosted in web, it so much vital to protect both the database and the valuable data containing it. Today, these kinds of databases are getting attacked frequently by the criminals due to its inadequate security mechanisms.

- Before moving on to the security mechanisms to be implemented in “Restaurant Management System” database, let’s discuss how actually databases which are hosted in web actually works.

▪ What it actually means by "Web Security of a Database"?

- In Web-based environments, 3-layer architecture is used in most of the times. This consists with a back-end and a front-end. In the back-end, there are 2 main servers.

  1) The Web Server
  2) DBMS / Database Server

- Currently, the web server hosts static web pages or scripts, which are server-side programs interacting with a DBMS in order to access the database and to provide dynamic and on-demand content.

- A server-side program (web application) stores a plain-text account in order to create connections to the database. So as a result, any person who has granted a physical access to the server's file system is capable of viewing the password used for the authentication to the DBMS. So as a result of this the account of a database user is publically accessible.

- If we go in to depth, the account is usually formed of a couple of username and password. This account is used for all the connections to the database, independently from the user profile. But in order to make this process simpler it is the web application that accesses the database, not the user. So it's actually impossible for the DBMS to know who is really using the server-side program, and to apply its own authorization mechanism.

- **Ex:-** An organization's Database server consists with 2 roles : Manager and Employee. So there should be a policy that an employee cannot access the content/data which can be accessible only by the manager. So the privileges should be limited for the sake of the security of the database.

▪ Security Mechanisms that should be implemented

➢ "Restaurant Management System" database consists with lot of personal and sensitive information about the restaurant and its food items, employees and customers. So it's a huge task and a responsibility to protect the privacy of all related things to the database. Our main target has to be protecting and securing the CIA Triad : confidentiality, integrity and availability. So if we are capable of protecting the CIA Triad, that means our database can stand strong against malware attacks, sql injection, DoS attacks and other kinds of exploitations.

➢ Now let's consider about the attacks and security countermeasures that should be implemented when hosting the "Restaurant Management System" database in web.

1) SQL Injection

▪ SQL injection is one of the most used and most common web based attack. To make a SQL injection attack work there should be a web application which uses a database. This attack can harm any website or web application which uses an SQL based database.

▪ Simply we can define SQL injection as a code injection technique. This Technique is used to execute SQL statements which are malicious. The attacker embeds a malicious code to the front end of the web application and which is then passed to the backend of the database. These malicious codes manipulate the database query to do something which is not supposed to do. If the attack gets successful this can

lead to deletion of confidential information, unauthorized access to the system and compromising Personal Computers (PCs) or even compromise the entire database. Authentication mechanisms get by passed by this kind of attacks and then the attacker can illegally retrieve the content of an entire database. This is simply an attack which can be used to take over database servers.

❖ How SQL Injections works

- Let's say an attacker tries to gain access to all data about a particular customer from the CUSTOMER database table. This attacker doesn't know a valid username and password. So to do this the attacker uses a malicious code "**1' OR '1' = '1**". One of the valid username and password is James and jamespas123 respectively. So the query for this will be,

```
SELECT *
FROM CUSTOMERS
WHERE Username = 'James' AND Password = 'jamespas123'
```

- But the attacker doesn't have a valid username and password. As a solution for the attacker enters **1' OR '1' = '1** for both username and password. The query with the malicious code will be,

```
SELECT *
FROM CUSTOMERS
WHERE Username = '1' OR '1' = '1' AND Password = '1' OR '1' = '1'
```

- Using this attacker will access data of any customer whose password is 1 or if 1=1. Since 1=1 condition is always true, using this malicious code the attacker can bypass the authentication process and get access to customer data.

❖ Countermeasures to prevent SQL Injection

1. Use parameterized queries

- Parameterized queries are considered one of the best ways to prevent SQL injection. In here a parameter is used instead of inserting values directly to the command. From this method, running of malicious queries at the backend which are harmful to the database will be prevented.
- Let's say the attacker enters 1' OR '1' = '1 as the input, then the parameterized query will search the table for a data that matches the entire string 1'OR '1' = '1'.

2. Validating user inputs

- The attackers enter the malicious code as user inputs. So first, an input validation should be done. This input validation is done to make sure that the user input matches the accepted type, accepted length and accepted format.

3. Keep database credentials separately and keep them in encrypted format

- When storing database credentials, it is better to store them in a separate file and make sure that it is kept encrypted. If these are followed the attacker won't benefit much from the attack.

4. Hiding information from error messages

- Attackers use error messages to get an idea about the database architecture. So we should make sure that the error message will disclose only the information which are necessary. It's better to display a generic error message which tells us something is wrong and tell the user to contact the technical support team if required.

2) Weak Audit Trail

- Audit trail is a record which is generated for each and every transaction. This record keeps information about the transaction. This information are who (user or the application program and a transaction number) has initiated, where (location of the user and/or the terminal) it has got initiated, when (time and date) it got initiated and what is the purpose.

- A proper audit trail should collect and archive the detailed records of the data which is stored in the database. One of the mistakes that organizations do is that thinking that that the audit trails which are built in are enough to help them to stay complaint and secure. The built in audit trails are considered as weak audit trails. These are considered as weak audit trails because the built in audit trails do not record the details which are necessary to support audit and also these weak audit trails consume more CPU and disk resources. Other than that these built in audit trails are unique for a particular database server platform, because of this the organization with different database environments may face obstacles when implementing uniform audit process.

- Now let's look at what is the problem with weak audit trails. Let's say ABC organization experiences a data breach. From this breach only 10% were actually affected. But we do not have a way to prove that not all customers got affected but only 10% got affected. This is the reason why proper auditing is required. Without proper auditing, you will not be able to get this kind of information. If there was a proper audit trail ABC organization can prove that only 10% got affected. Not only had this we can use a good audit trail to detect an active threat before it harmed the system.

❖ Countermeasures to prevent Weak Audit Trail

1. **Use network based audit appliances**

- These network based audit appliances do not have an impact on performance of the database and the appliances work independently from all the users and also this offers granular data collection.

3) Unmanaged sensitive data

- As we all know data is the most important asset in any organization or business. This data must be protected and need to be managed properly. This sensitive data should get managed in a way in which the data doesn't fall into an attacker's hand. If this sensitive data is not managed properly and not secured this data will get exposed to threats.

- Sensitive data is very important. If the sensitive data fall into the hands of an attacker, the attacker can do various things like identity theft, etc. Most of the organizations store bulk of data and they fail to manage them properly. Sometimes the forgotten and unattended data may fall into the target of hackers. Not only the forgotten data, but also the newly added data can fall to the hacker's prey. New data will be added to the database in a daily basis and because of the huge amount of data added regularly, it is difficult to track all of them, because of this the newly added data can get exposed to threats. So the organization which owns the database must manage the data properly. Otherwise it will lead to serious problems like data breaches, etc. The sensitive must not get exposed to attackers and the organizations which own the database must take the necessary steps to manage and secure the sensitive data properly.

❖ Countermeasures to prevent unmanaged sensitive data

1. Encrypt all sensitive data in your database

- Storing user passwords, credit card information or any other sensitive information in plain text is a huge problem. We should encrypt the sensitive data strongly and properly, because weak encryption is considered no better than no encryption.

- By encrypting, plain text which is in readable format is converted into cipher text which is in an unreadable format. Only the people who know the key will be able to unencrypt the encrypted data and read it. It is a must to use proper encryption mechanisms to encrypt sensitive data. If the data is encrypted, then only the authorized people will be able to read the encrypted data.

- Using this, sensitive data can be protected.

2. Apply proper database access controls

- Database access controls must be implemented in a correct way. Database access controls allows only the people who have the right to access the sensitive data to access it. Proper access controls should be implemented in a way in which only the required people have access to required data (no less or no more).

4) Inadequate database backups

- Database backups are actually one of the most important and a must done thing when it comes to databases, but unfortunately even though this is an important task most organizations have pushed the backup process to the bottom of the priority list.

- A particular organization will only realize the importance of backups only after something breaks down and critical data is lost. The data should be backed up regularly and the organizations must understand the importance of backing up data on a regular basis. Let's look at an example to understand the importance of database backups let's say data of a bank is lost due to a media failure and data don't have a backup copy. This might affect the entire bank's system, bank's reputation and also will affect customers in a huge way. But if data is backed up regularly this problem won't happen.

- Because of this data backups are considered as an important thing in an event of a recovery.

❖ Countermeasures to prevent Inadequate Database Backups

1. Back up data on a regular basis

- The data must be backed up on a regular basis. If data is backed up regularly, then every data will be backed up without missing certain data. When data is backed up regularly, the backup copy will contain all the data up to date. Because of this the organizations must understand the importance of backing up data in a regular basis.

2. Backing up all important data

- All the data should be backed up. So even if data is lost, a copy of every single data will be available in the backup copy. Because of this there won't be problems with gaining back the lost data.

5) Database Backups Exposure

- A database is a collection of sensitive data. Data storage refers to holding the data files in a secure location that can readily and easily access. Data backup, in contrast, refers to saving additional copies of the data in separate physical or virtual locations from data files in storage.

- The main reason for data backup is to save sensitive data if a system crash or hard drive failure occurs.

- Data exposure occurs as a result of not sufficiently protecting a database, such as weak encryption, no encryption, software flaws, or mistakenly uploads data to an incorrect database.

❖ Disadvantages of Database Backups Exposure:

a) Accidental or malicious damage/modification to data
b) Theft of valuable data
c) Breach of confidentiality agreements and privacy laws

❖ Countermeasures to prevent Database Backups Exposure

1. Encrypt all sensitive data in database.
2. Apply required controls and permissions to the database.
3. Run periodic search for new sensitive data on the databases. (Periodic Data Discovery tool and Compliance Manager that will automatically discover newly added sensitive data and protect it.)
4. Taking the appropriate measures to protect backup copies of sensitive data.

6) Denial of Service attack

- A DoS attack is one where the attacker does not manage to compromise a service, gain elevated privileges or steal information. Instead, of that attacker brings the service down or cripples it to a point that legitimate users of the service cannot use it effectively.

❖ Different classes of denial of service attacks:

1) Abuse of Functions
2) Complex Queries
3) Bugs and Defects
4) Application Usage

- A distributed denial-of-service (DDoS) attack occurs when multiple machines are operating together to attack one target. DDoS attackers often use the botnet (a group of hijacked internet-connected devices to carry out large scale attacks).

- Security vulnerabilities or device weaknesses are some disadvantages that attackers use to control numerous devices using command and control software. Once in control, an attacker can command their botnet to conduct DDoS on a target. In this case, the infected devices are also victims of the attack.

❖ Countermeasures to prevent Denial of Service Attack

1. Application and Database Abstraction Layer Hardening.
2. Apply Database Firewalls.
3. Configure network hardware against DDoS attacks.
4. Use a network Intrusion Detection System (IDS).
5. Employ dynamic backlog mechanisms to ensure that the connection queue is never exhausted.
6. Implementation of Intrusion Prevention Systems (IPS s) with DDoS detection capability.

7) Exploitation of Vulnerable Misconfigured Databases

- Misconfiguration is one of the top three vulnerabilities that lead to unauthorized access to databases. Attackers could benefit from unpatched databases or archives not properly configured that still have default accounts and configuration parameters. Such flaws frequently give attackers unauthorized access to some system data or functionality. Such flaws result in a complete system compromise. The first step for a penetration testing is the analysis of those flaws.

❖ Following vulnerabilities are the most critical ones related to database misconfiguration:

1) Misconfigured Privileges.
2) Unnecessary Access to Sensitive Data

- According to the 2014 Independent Oracle User Group (IOUG) Enterprise Data Security Survey, 36 percent of Oracle users take more than six months to apply a Critical Patch Update, while another 8 percent have never applied one.

❖ Countermeasures to prevent Exploitation of Vulnerable Misconfigured Databases

1. Use vulnerability assessment activities.
2. Use an efficient patch management process.
3. User accounts must be created using fresh username and password.

8) Excessive Database privileges

- One of a most important security method is to give least privileges to the users. Means a user should have no more access to data and system than is necessary for their task. Too often security problems result from this matter. Sometimes this can be carried out furthermore. Like privilege escalation and privilege abuse.

- For an example there was a story of Edward Snowden, NSA contractor. If even the national security agency (NSA), where "security" is in their middle name, doesn't take it seriously enough, how much effort is the average private company putting into privilege management? In 2013 Edward Joseph Snowden has copied and leaked highly classified information from the national security agency (NSA). This happened because of the excessive database privileges issue. BeyondTrust is a company, which makes software to assist companies in limiting user privilege, demonstrates that bad privilege habits run rampant. This company had done a survey about this National Security Agency matter. According to the survey they have find out some main points which need more attention.

- 44% of employees have access right that are not necessary to their current role.
- Customer information is considered most likely at risk is there's a lack of proper access controls over privileged users.

❖ Countermeasures to prevent Excessive Database privileges

1. Uphold a strict access and privileges control policy.
2. Don't grant excessive privileges to company employees and revoke outdated privileges in time.
3. Having a good policies and strict controls over the privileges and the system. Can avoid excessive database privilege.

9) Database Vulnerabilities and Misconfigurations

- Database vulnerability is a loophole which can be used to do an attack to the system.

1) Database privilege escalation
2) SQL Injection
3) Deployment failures

- Those are some of examples for Database vulnerabilities.

- Databases are found totally unprotected due to misconfiguration. Things like that can be happen in a process of creating a database. One should remember that hackers are often highly professional IT specialists who surely know how to exploit database vulnerability and misconfigurations and using them to attack to the database.

❖ Countermeasures to prevent Database Vulnerabilities and Misconfigurations

1. IT personnel should be highly qualified and experienced in the industry.
2. Avoid having any default accounts.
3. Encrypt the data.
4. Enforce password policies.
5. Secure your database using security methods.
6. Avoid common programming errors.
7. Keep your systems and applications updated

- These are some countermeasures that we can take to avoid database vulnerabilities.
- From having vulnerabilities in a system database can do serious damage to your operating system, server application and to the reputation.

10) Limited Security Expertise and Education

- Nowadays everything's are controls over the computers. Because of that there is a possibility that the system or the database can get attack from the attackers. To avoid that a company need persons who are professional in IT and IT security.

- Database can get breached and leaked due to insufficient level of it security expertise and education of non-technical employees who may break basic database security rules and put database at a risk. Because of that having an expertise is essential.

- When creating a database, it should create in the proper way with using security methods. If not, database could at a risk. To do so need database security expertise.

❖ Countermeasures to prevent Limited Security Expertise and Education

1. Database user must be educated in database security.
2. IT security specialists shall be urged to raise their professional level and qualification.
3. Always should keep them updated from technology.
4. Give trainings.

- By doing these can have a good expertise in the industry. Can ensure the protection of the database and the system.

Implementing Countermeasures and Additional Security Mechanisms

1) Transparent Data Encryption

```
-- 1) Transparent Data Encryption--

-- Setting up the master encryption key--
ALTER SYSTEM SET ENCRYPTION KEY IDENTIFIED BY "no_one_can_guess_this_#PASSWORD1";

-- Whenever the database is restarted, you have to load the master encryption key again--
ALTER SYSTEM SET ENCRYPTION WALLET OPEN IDENTIFIED BY "no_one_can_guess_this_#PASSWORD1";


-- Encryption of table columns--

-- User_account table--
ALTER TABLE User_account MODIFY (password VARCHAR(20) ENCRYPT, Username VARCHAR(50) ENCRYPT);

-- Employee_Phone table--
ALTER TABLE Employee_Phone MODIFY (Phone VARCHAR(50) ENCRYPT);

-- Employee_Email table--
ALTER TABLE Employee_Email MODIFY (Email VARCHAR(50) ENCRYPT);

-- Customer_Phone_number table--
ALTER TABLE Customer_Phone_number MODIFY (Phone_number VARCHAR(20) ENCRYPT);

-- Customer_C_Email table--
ALTER TABLE Customer_C_Email MODIFY (C_Email VARCHAR(70) ENCRYPT);

-- (Optional) Disabling the access to the master key--
ALTER SYSTEM SET ENCRYPTION WALLET CLOSE;
```

2) Stored Procedure

```
-- 2) Stored Procedure--

-- Payment table--

-- Compiling the procedure--

CREATE OR REPLACE PROCEDURE
    update_payment(P_ID IN VARCHAR, AMT IN REAL)         -- Setting up the parameteres--
IS
BEGIN
    UPDATE Payment
    SET Payment_amount = Payment_amount + AMT
    WHERE Payment_ID = P_ID;


END;
/


-- Executing the procedure--

EXEC update_payment('PN1010', 500);
```

```
-- Food_Item table--

-- Compiling the procedure--

CREATE OR REPLACE PROCEDURE
    update_food_price(I_ID IN VARCHAR, INCRMNT IN REAL)         -- Setting up the parameteres--
IS
BEGIN
    UPDATE Food_Item
    SET Item_price = Item_price + INCRMNT
    WHERE Item_ID = I_ID;


END;
/


-- Executing the procedure--

EXEC update_food_price('FID001', 40);
```

```
-- Employee table--

-- Compiling the procedure--

CREATE OR REPLACE PROCEDURE
    update_employee_city(E_ID IN VARCHAR, NWCTY IN VARCHAR)         -- Setting up the parameteres--
IS
BEGIN
    UPDATE Employee
    SET City = NWCTY
    WHERE Emp_ID = E_ID;


END;
/


-- Executing the procedure--

EXEC update_employee_city('M1234', 'Peradeniya');
```

```
-- Delivery_Person table--

-- Compiling the procedure--

CREATE OR REPLACE PROCEDURE
    update_license_state(E_ID IN VARCHAR, LCNS IN VARCHAR)         -- Setting up the parameteres--
IS
BEGIN
    UPDATE Delivery_Person
    SET License_state = LCNS
    WHERE Emp_ID = E_ID;


END;
/


-- Executing the procedure--

EXEC update_license_state('M3463', 'Valid');
```

3) View

```sql
-- 3) View--

-- Orders, Customer tables--

CREATE VIEW vw_orders AS
SELECT O.Order_ID, C.First_name, C.City, O.Order_Date, O.Delivery_Date, O.Total_price, O.Order_Status
FROM Orders O, Customer C
WHERE O.Customer_ID = C.Customer_ID;

SELECT *
FROM vw_orders;


-- Orders, Food_Item, Cart tables--

CREATE VIEW vw_delivery AS
SELECT O.Order_ID, C.Cart_Description, F.Item_name, F.Item_price, F.Item_quantity, O.Total_price, O.Order_Status, O.Delivery_Date
FROM Orders O, Food_Item F, Cart C
WHERE O.Order_ID = F.Order_ID AND C.Cart_ID = O.Cart_ID;

SELECT *
FROM vw_delivery;
```

```sql
-- Delivery_Person, Employee tables--

CREATE VIEW vw_delivery_person AS
SELECT D.Emp_ID, E.First_name, E.Last_name, E.House_number, E.Street, E.City, E.DOB
FROM Delivery_Person D, Employee E
WHERE D.Emp_ID = E.Emp_ID;

SELECT *
FROM vw_delivery_person;



-- Customer, Customer_Phone_number, Customer_C_Email tables--

CREATE VIEW vw_customer_contact AS
SELECT C.Customer_ID, C.First_name, P.Phone_number, E.C_Email
FROM Customer C, Customer_Phone_number P, Customer_C_Email E
WHERE C.Customer_ID = P.Customer_ID AND C.Customer_ID = E.Customer_ID;

SELECT *
FROM vw_customer_contact;
```

```sql
-- Orders table--

CREATE VIEW vw_orders AS
SELECT Order_ID, Order_Date, Order_Status, Total_price
FROM Orders

UPDATE vw_orders
SET Order_Status = 'Complete'
WHERE Order_ID = '004';
```

Database Recovery Mechanisms

- Database recovery is the process of restoring the database to the most recent consistent state that existed just before the failure.

- There are three states of database recovery.
    1) Pre- condition
    2) Condition
    3) Post- condition

- Pre-condition is any given time that the database works in a consistent state. Condition is when a system failure happened. Post-condition is restoring the database to the consistent state that existed before the failure.

➢ There are many ways to occur a system failure.

1) Transaction failure

i. Erroneous of parameter values.
ii. Logical programming errors
iii. System errors
iv. User interruption.
v. Concurrency control enforcement

2) Malicious transaction

3) System crash.

i. Hardware, software, or network error.

4) Disk crash

- There are three main types of recovery methods.

    1) Deferred update
    2) Immediate update
    3) Shadow paging

❖ Deferred Update

- During transaction execution, the updates are recorded only in the log and in the cache buffer. After the transaction reaches its commit point and the log is force-written to disk, the updates are recorded in the database. If the transaction fails before its reach commit point. There is no need to undo any operation because it's not recorded in the disk yet. So this may simplify the recovery process.

- Deferred update method is most suitable method for a restaurant. Because they are not handling any critical information. After every transaction reaches the commit point it can back up to the disk. Not only that, it can recovery everything from the backup also. So that it won't take a long time to backup than shadow paging recovery method.