Course: COMP1639

Database Engineering

Due Date: 20nd November 2017

Submitted to:	**Submitted by:**
Mr Andy Wicks	**Name:** Fatema Akter
Greenwich Course leader,	**ID:** 000990797
University of Greenwich, UK	

Date of Submission: 20nd November 2017

Topic Name: Restaurant Delivery Service

You tube Address: https://youtu.be/79IxLJOMK-Q

Table of Contents

Introduction: 4
Brief Description of the Database Assumption: 5
ERD (Entity Relationship Diagram): Given Below- 6
Mapping ERD to 3NF: 7
Database Statement Used Create Table: 8
"tbl_12hours_service" table statement 8
"tbl_24hours_service" table statement 9
"tbl_address" table statement 10
"tbl_branch" table statement 11
"tbl_customer" table statement 12
"tbl_day_shift" table statement 13
"tbl_delivery" table statement 14
"tbl_delivery_time_card" table statement 15
"tbl_employee" table statement 16
"tbl_night_shift" table statement 17
"tbl_restaurant" table statement 18
"tbl_rules" table statement 19
"tbl_service" table statement 20
Create Query Statement with Demonstration: 21
1. Query for Order By: 21
2. Query For Inner Joins: 22
3. Query for Sub-Query Where Clause: 23
4. Query For Partial Matching Where Clause: 24
5. Query for Aggregate Function: 25
6. Query For Grouped By and Having Clauses: 26
7. Query for Sub-Query as a Relation: 27
8. Query For Self Join: 28
9. Query For Create View: 29
10. Query for view as a relation: 31
11. Trigger: 32
Evaluation: 37

Conclusion:.....38
Self assessment:.....39
Bibliography 40

Introduction:

This assignment topic is independent for student. Which, I have chosen ***"restaurant delivery service"*** for designing my ***entity relationship diagram***. I have chosen this topic my own interest. My database mainly ***focuses on restaurant delivery service***.

Brief Description of the Database Assumption:

My chosen topic is ***"restaurant delivery service".*** I have to ***design and developed database*** with my chosen topic. I have ***selected my topic from URL*** which provide in Crouse work. In this course work I am ***visited many type of restaurant*** and ***searched different type of website***. Which I have ***gain different knowledge*** about restaurant delivery service. By monitoring restaurant physically and I have set some entity. Which I have used entity in the database system. I have to use many type of entities for developing database like- ***"tbl_12hours_service", "tbl_24hours_service", "tbl_address", "tbl_branch", "tbl_customer", "tbl_day_shift", "tbl_delivery", "tbl_delivery_time_card", "tbl_employee", "tbl_night_shift", "tbl_restaurant", "tbl_rules"*** and ***"tbl_service".*** I am going to design an entity relationship model for relational database by using all thirteen entities. I have described all the thirteen entities. ***Following chen's foot notation*** I am ***developed ERD*** for the ***restaurant delivery service***. Restaurant delivery service web performs where any customer can choice ***package by own choice***. There have ***many type of restaurant*** and they have ***different branch***. The branch provides ***service time 12hours and 24 hours***. The customer can choices any service by own choice. But the 12hours and 24hours service ***flow rules***. Both service times are under of rules. And they provide ***two types of service day shift and night shift.*** The customers can choice any time. Day service provides package a ***common price*** but night services provide ***package with extra charge*** .both are use for ***data redundancy***. The customers can payment by card for ***their delivery package***. They will be ***keeping some information like - branch, service, delivery and employee***. All the ***branch has own address***. All type of address is ***under of branch***. Which the "address" will be kept all type of addresses ***like-customer address and employee address and branch address*** .my database design by this process.

ERD (Entity Relationship Diagram): Given Below-

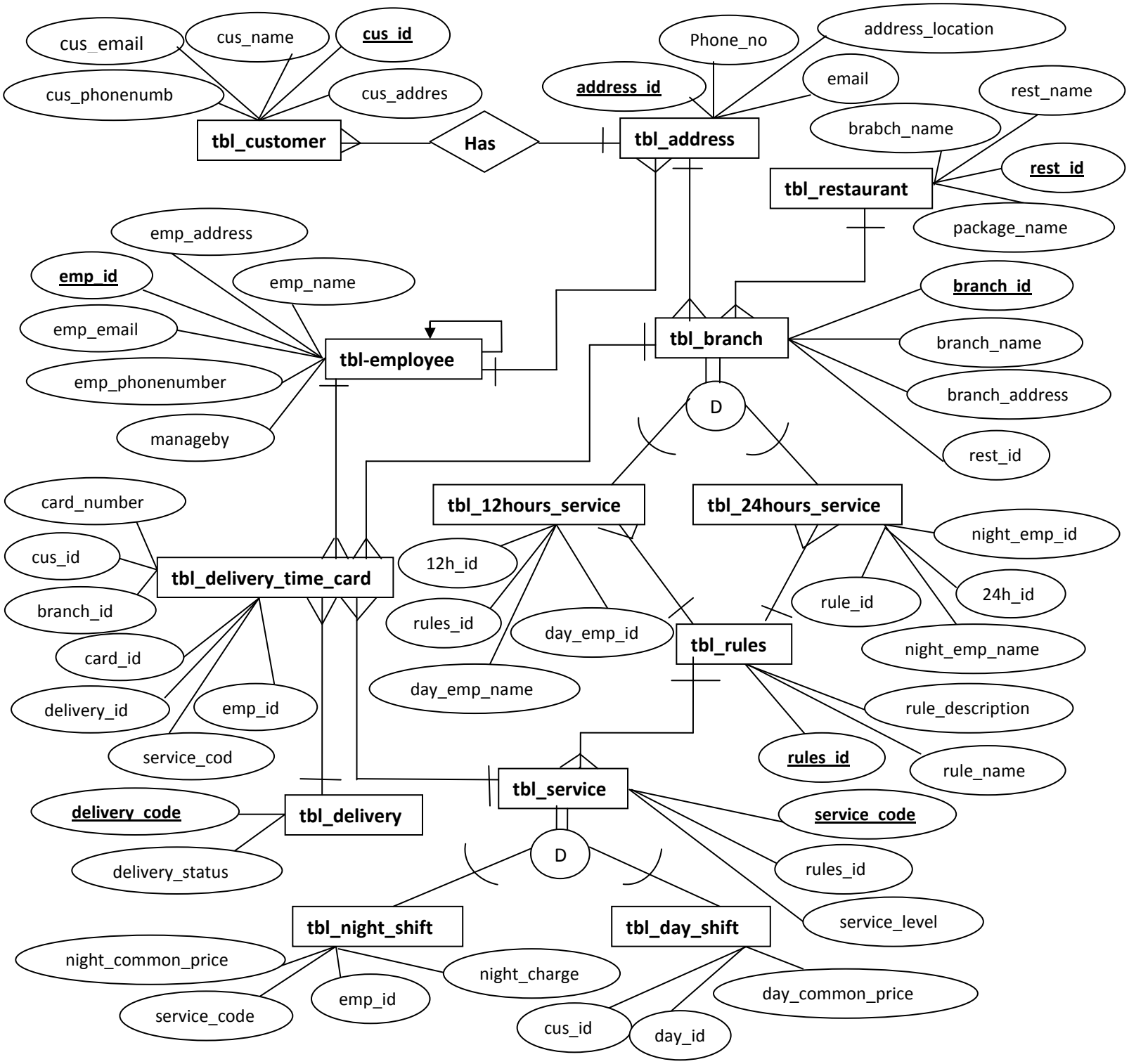


Figure: 1.1-Digram for ERD

Mapping ERD to 3NF:

I have developed ***Entity Relationship Model*** by using restaurant delivery service. For Develop this system used this ***relational data model***. All the relationship ***identified*** from this ***entity relationship data model.*** Which I have ***identified entity and attributes***. All the data help me mapping for entity relationship data model.

No	Table	Attributes
01	tbl_12hours_service	12h_id,day_emp_id,day_emp_name,**rules_id**
02	tbl_24hours_service	24h_id,night_emp_id,night_emp_name,**rules_id**
03	tbl_address	address_id, address_location, phone_no, email
04	tbl_branch	branch_id,branch_name,**branch_address,rest_id**
05	tbl_customer	cus_id,cus_name,**cus_address**,cus_phonenumber,cus_email
06	tbl_day_shift	day_id,cus_id,day_common_price
07	tbl_delivery	delivery_code,delivery_status
08	tbl_delivery_time_card	card_id,**emp_id,cus_id,service_code**,card_number,**delivery_id,branch_id**
09	tbl_employee	emp_id,emp_name,emp_address,emp_phonenumber,emp_email, **managedby**
10	tbl_night_shift	service_code,emp_id,night_common_price,night_charge
11	tbl_restaurant	rest_id,rest_name,branch_name,package_name
12	tbl_rules	rules_id,rule_name,rule_description
13	tbl_service	service_code,**rules_id,**service_level

Figure: 1.2-Table for ERD TO 3NF

I have map ***entity relationship data model using 3rd normalized from***. By the requirement I have mapped database using 3rd normalization from. So relational data model ***maintains 3rd normalized from.***

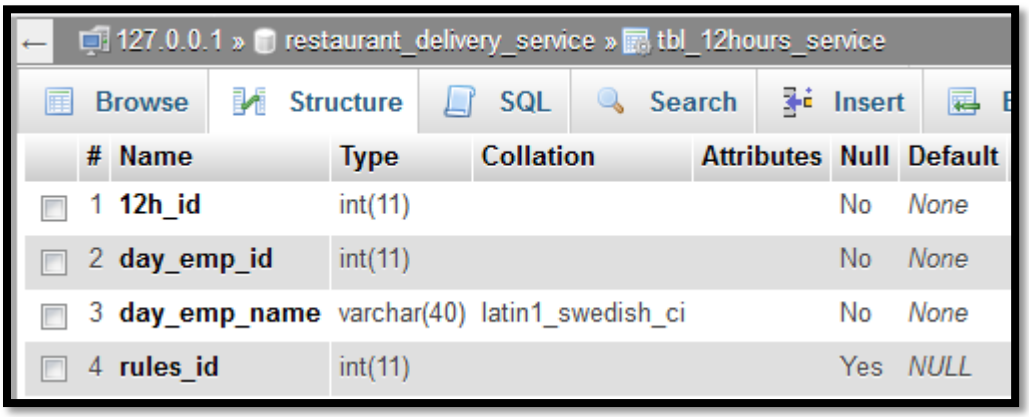
Database Statement Used Create Table:

"tbl_12hours_service" table statement

SQL Code for table "tbl_12hours_service":

```
CREATE TABLE IF NOT EXISTS `tbl_12hours_service` (

  `12h_id` int(11) NOT NULL,

  `day_emp_id` int(11) NOT NULL,

  `day_emp_name` varchar(40) NOT NULL,

  `rules_id` int(11) DEFAULT NULL,

  KEY `rules_id` (`rules_id`) USING BTREE

) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;
```

Table output:



#	Name	Type	Collation	Attributes	Null	Default
1	12h_id	int(11)			No	None
2	day_emp_id	int(11)			No	None
3	day_emp_name	varchar(40)	latin1_swedish_ci		No	None
4	rules_id	int(11)			Yes	NULL

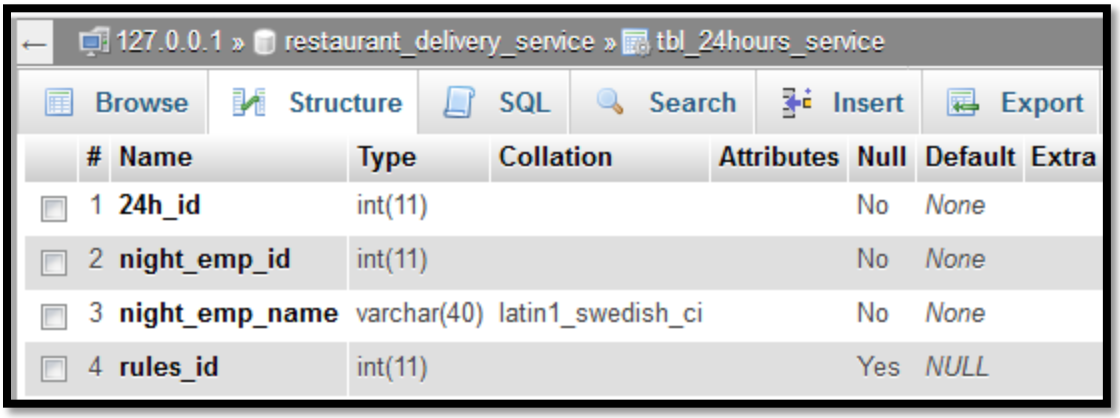
Figure: 1.3-Table for "tbl_12hours_service"

“tbl_24hours_service” table statement

SQL Code for table “tbl_24hours_service”:

```
CREATE TABLE IF NOT EXISTS `tbl_24hours_service` (

  `24h_id` int(11) NOT NULL,

  `night_emp_id` int(11) NOT NULL,

  `night_emp_name` varchar(40) NOT NULL,

  `rules_id` int(11) DEFAULT NULL,

  KEY `rules_id` (`rules_id`) USING BTREE

) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;
```

Table Output:



#	Name	Type	Collation	Attributes	Null	Default	Extra
1	**24h_id**	int(11)			No	*None*	
2	**night_emp_id**	int(11)			No	*None*	
3	**night_emp_name**	varchar(40)	latin1_swedish_ci		No	*None*	
4	**rules_id**	int(11)			Yes	*NULL*	

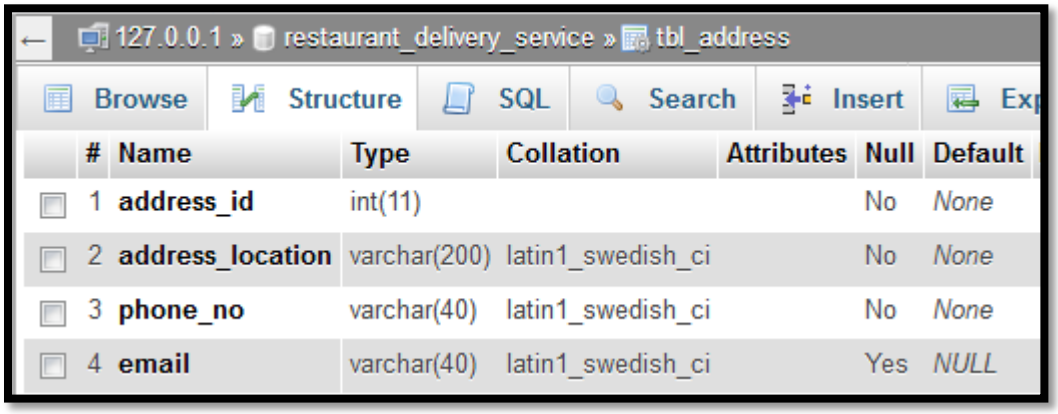
Figure: 1.4-Table for “tb_24hours_service”

“tbl_address” table statement

SQL Code for table “tbl_address”:

```
CREATE TABLE IF NOT EXISTS `tbl_address` (

  `address_id` int(11) NOT NULL,

  `address_location` varchar(200) NOT NULL,

  `phone_no` varchar(40) NOT NULL,

  `email` varchar(40) DEFAULT NULL,

  KEY `cus_id` (`address_id`) USING BTREE

) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;
```

Table Output:



#	Name	Type	Collation	Attributes	Null	Default
1	address_id	int(11)			No	None
2	address_location	varchar(200)	latin1_swedish_ci		No	None
3	phone_no	varchar(40)	latin1_swedish_ci		No	None
4	email	varchar(40)	latin1_swedish_ci		Yes	NULL

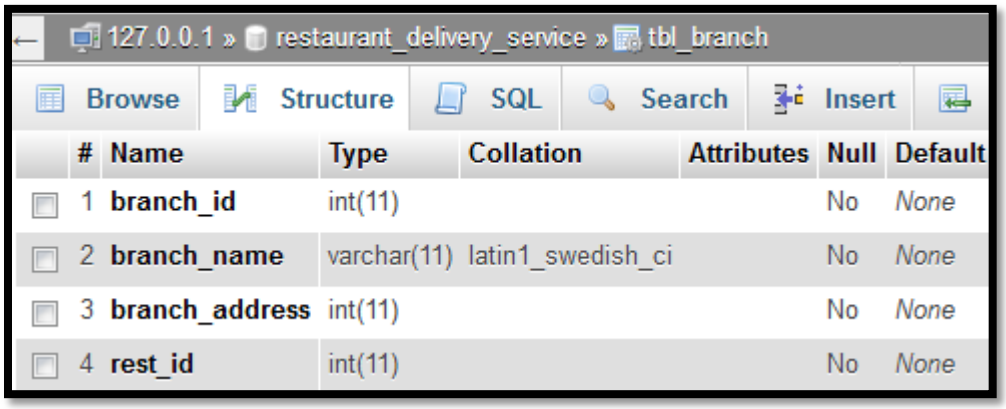
Figure: 1.5-Table for “tbl_address”

“tbl_branch” table statement

SQL Code for table “tbl_branch”:

```
CREATE TABLE IF NOT EXISTS `tbl_branch` (

  `branch_id` int(11) NOT NULL,

  `branch_name` varchar(11) NOT NULL,

  `branch_address` int(11) NOT NULL,

  `rest_id` int(11) NOT NULL,

  KEY `branch_address` (`branch_address`) USING BTREE,

  KEY `rest_id` (`rest_id`) USING BTREE,

  KEY `branch_id` (`branch_id`) USING BTREE

) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;
```

Table Output:



#	Name	Type	Collation	Attributes	Null	Default
1	branch_id	int(11)			No	None
2	branch_name	varchar(11)	latin1_swedish_ci		No	None
3	branch_address	int(11)			No	None
4	rest_id	int(11)			No	None

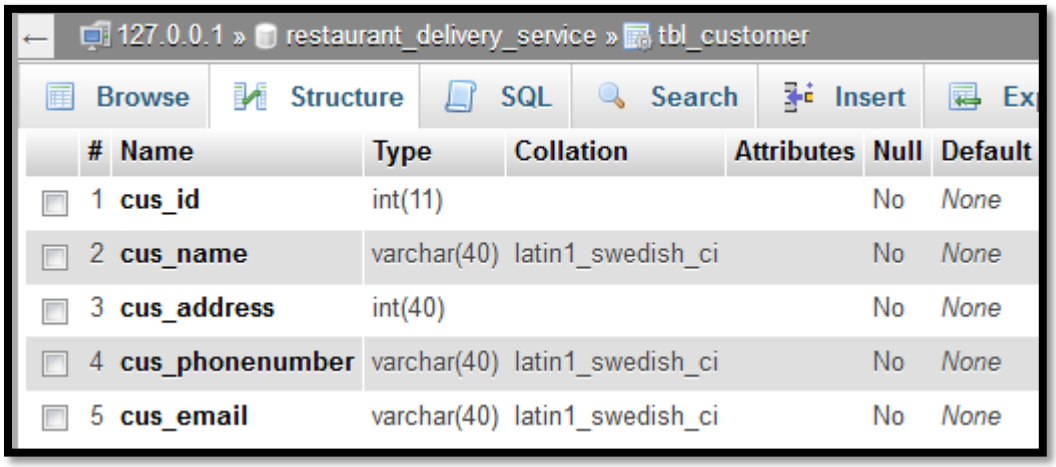
Figure: 1.6-Table for “tbl_branch”

“tbl_customer” table statement

SQL Code for table “tbl_customer”:

```
CREATE TABLE IF NOT EXISTS `tbl_customer` (

  `cus_id` int(11) NOT NULL,

  `cus_name` varchar(40) NOT NULL,

  `cus_address` int(40) NOT NULL,

  `cus_phonenumber` varchar(40) NOT NULL,

  `cus_email` varchar(40) NOT NULL,

  KEY `cus_address` (`cus_address`) USING BTREE,

  KEY `cus_id` (`cus_id`) USING BTREE

) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;
```

Table Output:



#	Name	Type	Collation	Attributes	Null	Default
1	cus_id	int(11)			No	None
2	cus_name	varchar(40)	latin1_swedish_ci		No	None
3	cus_address	int(40)			No	None
4	cus_phonenumber	varchar(40)	latin1_swedish_ci		No	None
5	cus_email	varchar(40)	latin1_swedish_ci		No	None

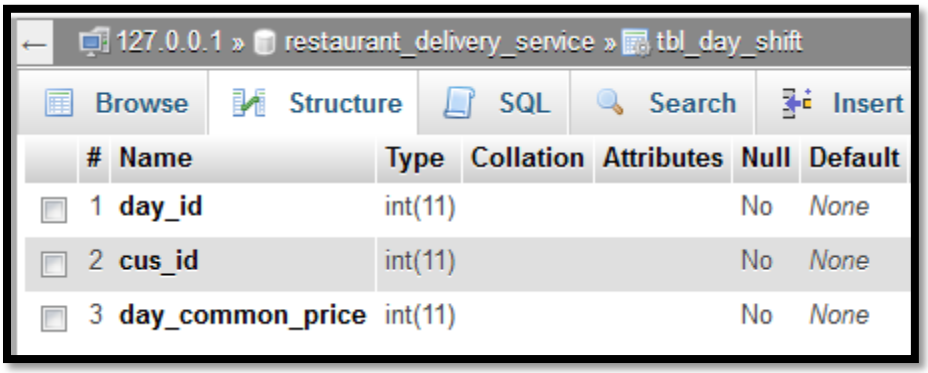
Figure: 1.7-table for “tbl_customer”

“tbl_day_shift” table statement

SQL Code for table “tbl_day_shift”:

```
CREATE TABLE IF NOT EXISTS `tbl_day_shift` (

  `day_id` int(11) NOT NULL,

  `cus_id` int(11) NOT NULL,

  `day_common_price` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;
```

Table Output:



#	Name	Type	Collation	Attributes	Null	Default
1	day_id	int(11)			No	None
2	cus_id	int(11)			No	None
3	day_common_price	int(11)			No	None

Figure: 1.8-table for tbl_day_shift

(Anon., n.d.)

“tbl_delivery” table statement

SQL Code for table “tbl_delivery”:

```
CREATE TABLE IF NOT EXISTS `tbl_delivery` (

  `delivery_code` int(11) NOT NULL,

  `delivery_status` varchar(255) DEFAULT NULL,

  KEY `delivery_code` (`delivery_code`) USING BTREE

) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;
```

Table Output:

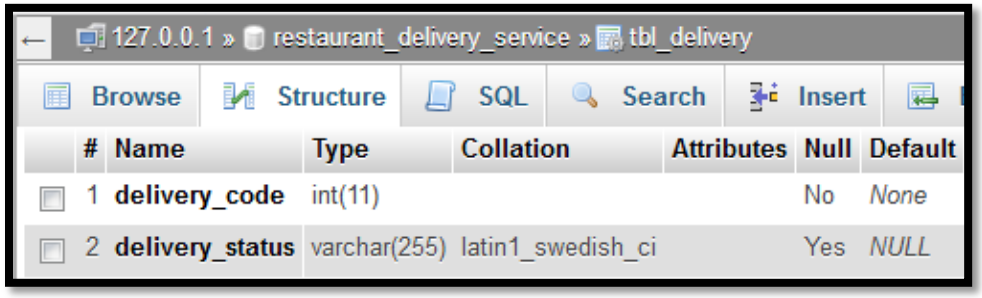


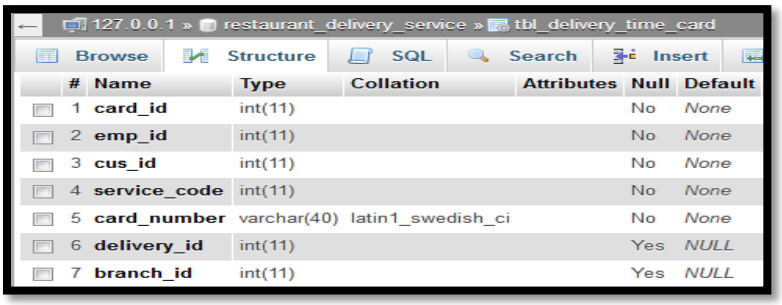
Figure: 1.9-table for “tbl_delivery”

“tbl_delivery_time_card” table statement

SQL Code for table “tbl_delivery_time_card”:

```
CREATE TABLE IF NOT EXISTS `tbl_delivery_time_card` (
  `card_id` int(11) NOT NULL,
  `emp_id` int(11) NOT NULL,
  `cus_id` int(11) NOT NULL,
  `service_code` int(11) NOT NULL,
  `card_number` varchar(40) NOT NULL,
  `delivery_id` int(11) DEFAULT NULL,
  `branch_id` int(11) DEFAULT NULL,
  KEY `emp_id` (`emp_id`) USING BTREE,
  KEY `cus_id` (`cus_id`) USING BTREE,
  KEY `service_code` (`service_code`) USING BTREE,
  KEY `delivery_id` (`delivery_id`) USING BTREE,
  KEY `branch_id` (`branch_id`) USING BTREE
) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;
```

Table Output:



127.0.0.1 » restaurant_delivery_service » tbl_delivery_time_card

Browse | Structure | SQL | Search | Insert

#	Name	Type	Collation	Attributes	Null	Default
1	card_id	int(11)			No	None
2	emp_id	int(11)			No	None
3	cus_id	int(11)			No	None
4	service_code	int(11)			No	None
5	card_number	varchar(40)	latin1_swedish_ci		No	None
6	delivery_id	int(11)			Yes	NULL
7	branch_id	int(11)			Yes	NULL

Figure: 1.10-Table for “tbl_delivery_time_card”

“tbl_employee” table statement

SQL Code for table “tbl_employee”:

```
CREATE TABLE IF NOT EXISTS `tbl_employee` (
  `emp_id` int(11) NOT NULL,
  `emp_name` varchar(40) NOT NULL,
  `emp_address` int(40) NOT NULL,
  `emp_phonenumber` varchar(40) NOT NULL,
  `emp_email` varchar(40) NOT NULL,
  `managedby` int(40) DEFAULT NULL,
  PRIMARY KEY (`emp_id`),
  KEY `managedby` (`managedby`) USING BTREE COMMENT 'self join column',
  KEY `emp_id` (`emp_id`) USING BTREE,
  KEY `emp_address` (`emp_address`) USING BTREE
) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;
```

Table Output:

127.0.0.1 » restaurant_delivery_service » tbl_employee

Browse | Structure | SQL | Search | Insert | Exp

#	Name	Type	Collation	Attributes	Null	Default
1	emp_id	int(11)			No	None
2	emp_name	varchar(40)	latin1_swedish_ci		No	None
3	emp_address	int(40)			No	None
4	emp_phonenumber	varchar(40)	latin1_swedish_ci		No	None
5	emp_email	varchar(40)	latin1_swedish_ci		No	None
6	managedby	int(40)			Yes	NULL

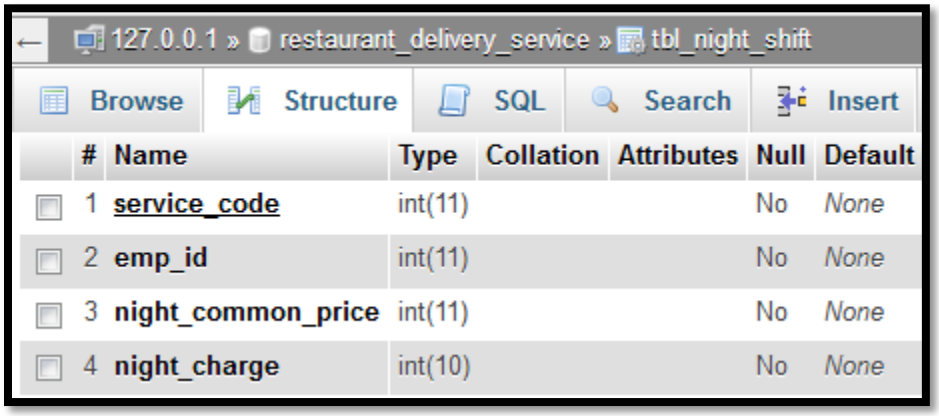
Figure: 1.11-Table for “tbl_employee”

“tbl_night_shift” table statement

SQL Code for table “tbl_night_shift”:

```
CREATE TABLE IF NOT EXISTS `tbl_night_shift` (

  `service_code` int(11) NOT NULL,

  `emp_id` int(11) NOT NULL,

  `night_common_price` int(11) NOT NULL,

  `night_charge` int(10) NOT NULL,

  PRIMARY KEY (`service_code`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;
```

Table Output:



#	Name	Type	Collation	Attributes	Null	Default
1	service_code	int(11)			No	None
2	emp_id	int(11)			No	None
3	night_common_price	int(11)			No	None
4	night_charge	int(10)			No	None

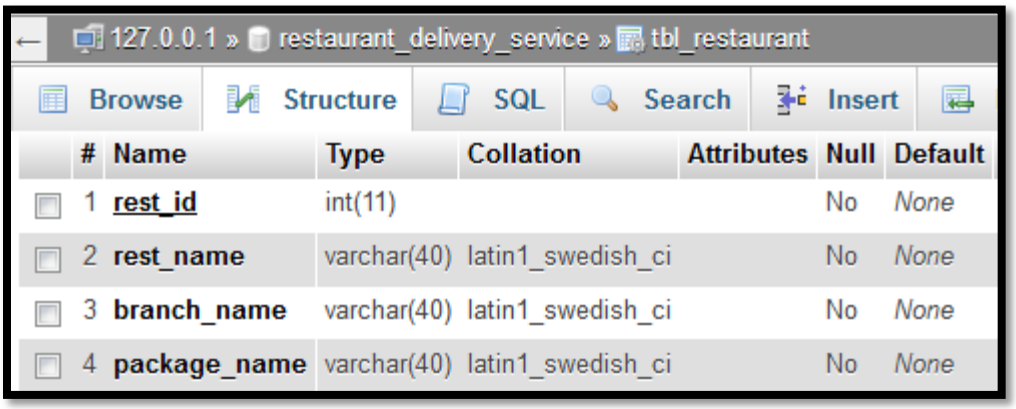
Figure: 1.12-Table for “tbl_night_shift”

“tbl_restaurant” table statement

SQL Code for table “tbl_restaurant”:

```
CREATE TABLE IF NOT EXISTS `tbl_restaurant` (

  `rest_id` int(11) NOT NULL,

  `rest_name` varchar(40) NOT NULL,

  `branch_name` varchar(40) NOT NULL,

  `package_name` varchar(40) NOT NULL,

  PRIMARY KEY (`rest_id`),

  KEY `rest_id` (`rest_id`) USING BTREE

) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;
```

Table Output:



#	Name	Type	Collation	Attributes	Null	Default
1	rest_id	int(11)			No	None
2	rest_name	varchar(40)	latin1_swedish_ci		No	None
3	branch_name	varchar(40)	latin1_swedish_ci		No	None
4	package_name	varchar(40)	latin1_swedish_ci		No	None

Figure: 1.13-Table for “tbl_restaurent”

“tbl_rules” table statement

SQL Code for table “tbl_rules”:

```
CREATE TABLE IF NOT EXISTS `tbl_rules` (

  `rules_id` int(11) NOT NULL,

  `rule_name` varchar(100) NOT NULL,

  `rule_description` text NOT NULL,

  KEY `rules_id` (`rules_id`) USING BTREE

) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;
```

Table Output:

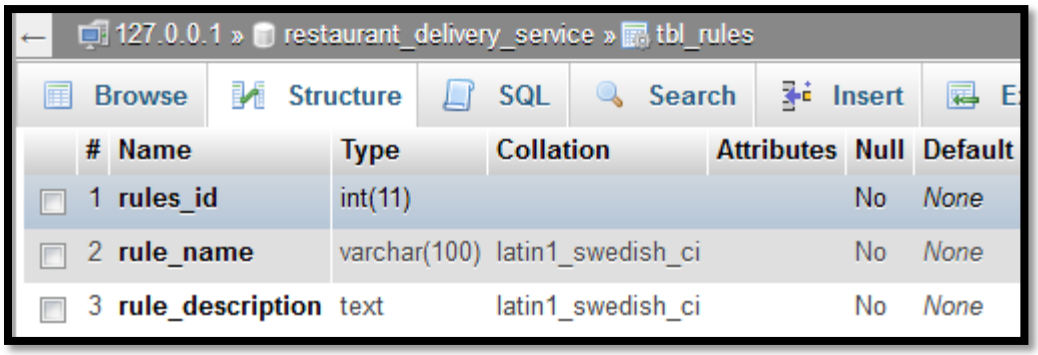


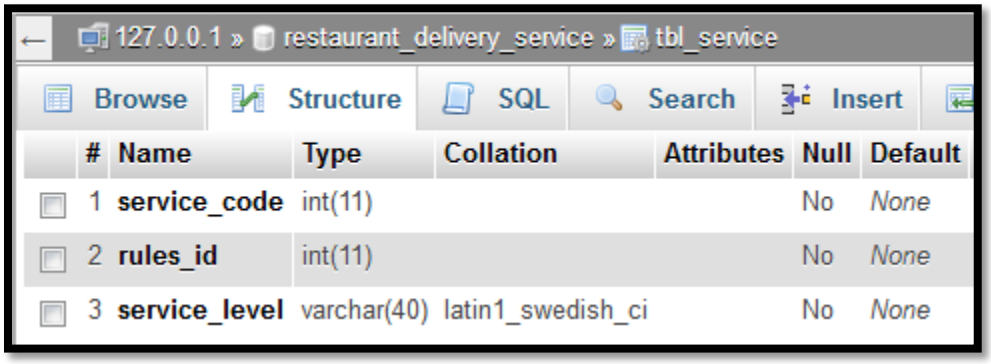
Figure: 1.14-Table for “tbl_rules”

“tbl_service” table statement

SQL Code for table “tbl_service”:

```
CREATE TABLE IF NOT EXISTS `tbl_service` (

  `service_code` int(11) NOT NULL,

  `rules_id` int(11) NOT NULL,

  `service_level` varchar(40) NOT NULL,

  KEY `rules_id` (`rules_id`) USING BTREE,

  KEY `service_code` (`service_code`) USING BTREE

) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;
```

Table Output:



#	Name	Type	Collation	Attributes	Null	Default
1	service_code	int(11)			No	None
2	rules_id	int(11)			No	None
3	service_level	varchar(40)	latin1_swedish_ci		No	None

Figure: 1.15-Table for “tbl_service”

(Anon., n.d.)

Create Query Statement with Demonstration:

1. Query for Order By:

Description:

This query for showing ***"order by" function,*** which show from ***"tbl_customer" table cus_id,cus-name and cus_address***.and then order by ***"cus_name" dese.***

Code for "Order By":

```
SELECT cus_id, cus_name, cus_address
from tbl_customer
order by cus_name desc;
```

Result: Query run successfully.

```
SELECT cus_id, cus_name, cus_address
FROM tbl_customer
ORDER BY cus_name DESC
LIMIT 0 , 30
```

Figure: 1.16-Display of "order by" query run successfully

Output:

+ Options

cus_id	cus_name	cus_address
601	Robert	401
606	Olivia	406
605	Emily	405
604	Bonny	404
602	Ava	402
603	Amelia	403

Figure: 1.17- Query output of "order by"

2. Query For Inner Joins:

Description:

This query is showing for ***"Inner Joins" function,*** which shows from ***"tbl_restaurant" table tbl_branch,*** which ***"inner joins" is rest_id,rest_name*** and ***branch_id, branch_name.***

Code for "Inner Joins":

SELECT r.rest_id,rest_name,tbl_branch.branch_id,tbl_branch.branch_name

FROM tbl_restaurant r

INNER JOIN tbl_branch ON r.rest_id=tbl_branch.rest_id;

Result: Query run successfully.

```
SELECT r.rest_id, rest_name, tbl_branch.branch_id, tbl_branch.branch_name
FROM tbl_restaurant r
INNER JOIN tbl_branch ON r.rest_id = tbl_branch.rest_id
LIMIT 0 , 30
```

Figure: 1.18- Display of "Inner Joins" query run successfully

Output:

+ Options

rest_id	rest_name	branch_id	branch_name
1	kfc	1301	Aaron's Hil
2	BFC	1302	Babbacombe

Figure: 1.19- Query output of "Inner Joins"

3. Query for Sub-Query Where Clause:

Description:

This query is showing for ***"Sub-Query Where Clause" function,*** which shows from ***"tbl_branch"*** table and which ***"sub- querry" is rest_id in rest_id.***

Code for "Sub-Query Where Clause":

```
SELECT *
FROM `tbl_branch`
WHERE rest_id in (SELECT rest_id FROM `tbl_restaurant` WHERE rest_name='BFC')
```

Result: Query run successfully.

```
SELECT *
FROM `tbl_branch`
WHERE rest_id
IN (

  SELECT rest_id
  FROM `tbl_restaurant`
  WHERE rest_name = 'BFC'
)
LIMIT 0 , 30
```

Figure: 1.20- Display of "sub-query where clause" query run successfully

Output:

+ Options

branch_id	branch_name	branch_address	rest_id
1302	Babbacombe	402	2

Figure: 1.21- Query output of "sub-query where clause"

4. Query For Partial Matching Where Clause:

Description:

This query is showing for ***"Partial Matching"*** function, which shows from ***"tbl_employee"*** table which ***"emp_email" like "yahoo".***

Code for "Partial Matching":

```
SELECT *
FROM `tbl_employee`
WHERE emp_email LIKE '%yahoo%'
```

Result: Query run successfully.

```
SELECT *
FROM `tbl_employee`
WHERE emp_email LIKE '%yahoo%'
LIMIT 0 , 30
```

Figure: 1.22- Display of "partial matching" query run successfully

Output:

+ Options

	emp_id	emp_name	emp_address	emp_phonenumber	emp_email	managedby
Edit Copy Delete	701	Isla	401	07074312345	isla@yahoo.com	NULL

Figure: 1.23- Query output of "particle matching

5. Query for Aggregate Function:

Description:

This query is showing for ***"Aggregate Function"*** which shows from ***"tbl_night_shift"*** table and showing night ***"average charge".***

Code for "Aggregate Function":

```
SELECT avg( night_charge ) AS 'average charge'
FROM `tbl_night_shift`
```

Result: Query run successfully.

```
SELECT AVG( night_charge ) AS 'average charge'
FROM `tbl_night_shift`
```

Figure: 1.24- Display of "aggregate function" query run successfully

Output:

average charge
13.7500

Figure: 1.25- Query output of "aggregate function"

6. Query For Grouped By and Having Clauses:

Description:

This query is showing for ***"Grouped By and Having Clauses"*** which shows from ***"tbl_night_shift"*** table and showing group by ***night_common_price*** and having ***"sum (night_charge)".***

Code for "Grouped By and Having Clauses":

```
SELECT night_common_price,sum(night_charge)
FROM `tbl_night_shift`
GROUP by night_common_price
HAVING sum(night_charge)>15
```

(Anon., n.d.)

Result: Query run successfully.

```
SELECT night_common_price, SUM( night_charge )
FROM `tbl_night_shift`
GROUP BY night_common_price
HAVING SUM( night_charge ) >15
LIMIT 0 , 30
```

Figure: 1.26- Display of "grouped by and having clauses" query run successfully

Output:

+ Options

night_common_price	sum(night_charge)
15	20
25	35

Figure: 1.27- Query output of "grouped by and having clause"

7. Query for Sub-Query as a Relation:

Description:

This query is showing for ***"Sub Query as a Relation"*** which shows from ***"tbl_restaurant"*** table and inner join from ***"tbl_branch"*** which showing ***rest_id, rest_name*** and ***branch_id, branch_name.***

Code for "Sub-Query as a Relation":

```
SELECT r.rest_id,rest_name,tbl_branch.branch_id,tbl_branch.branch_name

FROM tbl_restaurant r

INNER JOIN tbl_branch ON r.rest_id=tbl_branch.rest_id

and tbl_branch.branch_address = (select address_id from tbl_address where
address_location='27 Colmore Row Birmingham England B3 2EW')
```

Result:

```
SELECT r.rest_id, rest_name, tbl_branch.branch_id, tbl_branch.branch_name
FROM tbl_restaurant r
INNER JOIN tbl_branch ON r.rest_id = tbl_branch.rest_id
AND tbl_branch.branch_address = (
SELECT address_id
FROM tbl_address
WHERE address_location = '27 Colmore Row Birmingham England B3 2EW' )
LIMIT 0 , 30
```

Figure: 1.28- Display of "sub-query as a relation" query run successfully

Output:

+ Options

rest_id	rest_name	branch_id	branch_name
1	kfc	1301	Aaron's Hil

Figure: 1.29- Query output of "sub-query as a relation"

8. Query For Self Join:

Description:

This query is showing for ***"Self Join"*** function**,** which shows from ***"tbl_restaurant"*** table and ***"tbl_branch"*** table .where ***rest_is, rest_name*** and ***branch_is,branch_name.***

Code for "Self Join":

SELECT r.rest_id,rest_name,tbl_branch.branch_id,tbl_branch.branch_name

FROM tbl_restaurant r, tbl_branch

WHERE r.rest_id=tbl_branch.rest_id;

Result:

```
SELECT r.rest_id, rest_name, tbl_branch.branch_id, tbl_branch.branch_name
FROM tbl_restaurant r, tbl_branch
WHERE r.rest_id = tbl_branch.rest_id
LIMIT 0 , 30
```

Figure: 1.30- Display of "self join" query run successfully

Output:

+ Options

rest_id	rest_name	branch_id	branch_name
1	kfc	1301	Aaron's Hil
2	BFC	1302	Babbacombe

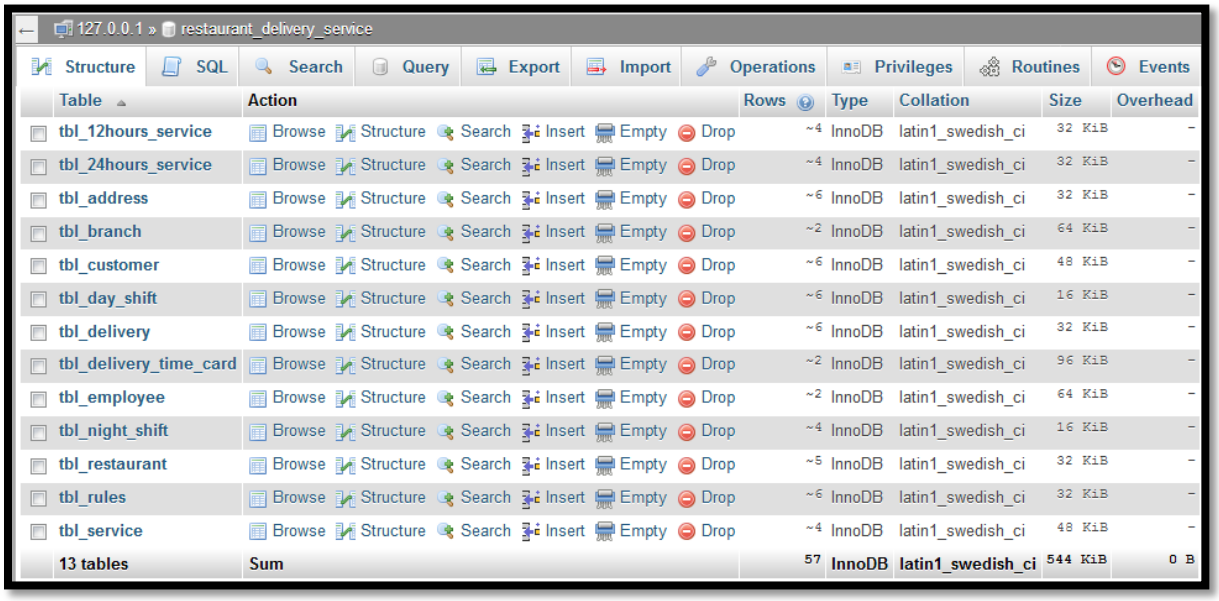
Figure: 1.31- Query output of "self join"

9. Query For Create View:

Description:

This query is showing for ***"Create View"*** which shows view table ***"branch_view***". From ***"tbl_branch"*** where select ***branch_id***, ***branch_name, address_location*** as address.

Table for Before Create View:



127.0.0.1 » restaurant_delivery_service

Structure | SQL | Search | Query | Export | Import | Operations | Privileges | Routines | Events

Table	Action	Rows	Type	Collation	Size	Overhead
tbl_12hours_service	Browse Structure Search Insert Empty Drop	~4	InnoDB	latin1_swedish_ci	32 KiB	-
tbl_24hours_service	Browse Structure Search Insert Empty Drop	~4	InnoDB	latin1_swedish_ci	32 KiB	-
tbl_address	Browse Structure Search Insert Empty Drop	~6	InnoDB	latin1_swedish_ci	32 KiB	-
tbl_branch	Browse Structure Search Insert Empty Drop	~2	InnoDB	latin1_swedish_ci	64 KiB	-
tbl_customer	Browse Structure Search Insert Empty Drop	~6	InnoDB	latin1_swedish_ci	48 KiB	-
tbl_day_shift	Browse Structure Search Insert Empty Drop	~6	InnoDB	latin1_swedish_ci	16 KiB	-
tbl_delivery	Browse Structure Search Insert Empty Drop	~6	InnoDB	latin1_swedish_ci	32 KiB	-
tbl_delivery_time_card	Browse Structure Search Insert Empty Drop	~2	InnoDB	latin1_swedish_ci	96 KiB	-
tbl_employee	Browse Structure Search Insert Empty Drop	~2	InnoDB	latin1_swedish_ci	64 KiB	-
tbl_night_shift	Browse Structure Search Insert Empty Drop	~4	InnoDB	latin1_swedish_ci	16 KiB	-
tbl_restaurant	Browse Structure Search Insert Empty Drop	~5	InnoDB	latin1_swedish_ci	32 KiB	-
tbl_rules	Browse Structure Search Insert Empty Drop	~6	InnoDB	latin1_swedish_ci	32 KiB	-
tbl_service	Browse Structure Search Insert Empty Drop	~4	InnoDB	latin1_swedish_ci	48 KiB	-
13 tables	Sum	57	InnoDB	latin1_swedish_ci	544 KiB	0 B

Figure: 1.32-Table for before create view table

Code for "Create View":

```
CREATE VIEW branch_veiw as

SELECT branch_id,branch_name,address_location as address,rest_id

FROM tbl_branch,tbl_address

where address_id=branch_address;
```

(Anon., n.d.)

Result:

```
CREATE VIEW branch_veiw AS SELECT branch_id, branch_name, address_location AS address, rest_id
FROM tbl_branch, tbl_address
WHERE address_id = branch_address
```

Figure: 1.33- Display of "create view" query run successfully

Output: After create view table, which table ***name is "branch_view"***

Structure | SQL | Search | Query | Export | Import | Operations | Privileges | Routines | Events

Table	Action	Rows	Type	Collation	Size	Overhead
branch_veiw	Browse Structure Search Insert Drop	~0	View	---	-	-
tbl_12hours_service	Browse Structure Search Insert Empty Drop	~4	InnoDB	latin1_swedish_ci	32 KiB	-
tbl_24hours_service	Browse Structure Search Insert Empty Drop	~4	InnoDB	latin1_swedish_ci	32 KiB	-
tbl_address	Browse Structure Search Insert Empty Drop	~6	InnoDB	latin1_swedish_ci	32 KiB	-
tbl_branch	Browse Structure Search Insert Empty Drop	~2	InnoDB	latin1_swedish_ci	64 KiB	-
tbl_customer	Browse Structure Search Insert Empty Drop	~7	InnoDB	latin1_swedish_ci	48 KiB	-
tbl_day_shift	Browse Structure Search Insert Empty Drop	~6	InnoDB	latin1_swedish_ci	16 KiB	-
tbl_delivery	Browse Structure Search Insert Empty Drop	~6	InnoDB	latin1_swedish_ci	32 KiB	-
tbl_delivery_time_card	Browse Structure Search Insert Empty Drop	~2	InnoDB	latin1_swedish_ci	96 KiB	-
tbl_employee	Browse Structure Search Insert Empty Drop	~2	InnoDB	latin1_swedish_ci	64 KiB	-
tbl_night_shift	Browse Structure Search Insert Empty Drop	~4	InnoDB	latin1_swedish_ci	16 KiB	-
tbl_restaurant	Browse Structure Search Insert Empty Drop	~5	InnoDB	latin1_swedish_ci	32 KiB	-
tbl_rules	Browse Structure Search Insert Empty Drop	~6	InnoDB	latin1_swedish_ci	32 KiB	-
tbl_service	Browse Structure Search Insert Empty Drop	~4	InnoDB	latin1_swedish_ci	48 KiB	-
14 tables	Sum	58	InnoDB	latin1_swedish_ci	544 KiB	0 B

Figure: 1.34-Display create view table of "branch_view"

10. Query for view as a relation:

Deception: This query is showing for ***"View as a Relation"*** where relation rest_id with rest_id from ***"branch_view***" and "***tbl_restaurant".***

Code for "View as a Relation":

```
SELECT b.rest_id,r.rest_name,branch_id,b.branch_name

FROM `branch_veiw` b, tbl_restaurant r

WHERE r.rest_id=b.rest_id;
```

Result:

```
SELECT b.rest_id, r.rest_name, branch_id, b.branch_name
FROM `branch_veiw` b, tbl_restaurant r
WHERE r.rest_id = b.rest_id
LIMIT 0 , 30
```

Figure: 1.35- Display of "view as a relation" query run successfully

Output:

+ Options

rest_id	rest_name	branch_id	branch_name
1	kfc	1301	Aaron's Hil
2	BFC	1302	Babbacombe

Figure: 1.36- Query output of "view as a relation"

11. Trigger:

Deception: Here trigger table is ***"tbl_customer",*** which "cus_phonenumber" ***start value like "01".*** If "cus_phonenumber" does not flow this rules then the ***"cus_phonenumber" is not valid*** is showing.

Table for Before Create Trigger:

+ Options

cus_id	cus_name	cus_address	cus_phonenumber	cus_email
601	Robert	401	07911177233	robert@gmail.com
602	Ava	402	07911177211	ava@gmail.com
603	Amelia	403	07911177280	amelia@gmail.com
604	Bonny	404	07911177555	bonny@gmail.com
605	Emily	405	07934721000	emily@gmail.com
606	Olivia	406	07911177555	olivia@gmail.com

Figure: 1.37-Display of before create trigger

Display of Trigger for "tbl_customer" table:

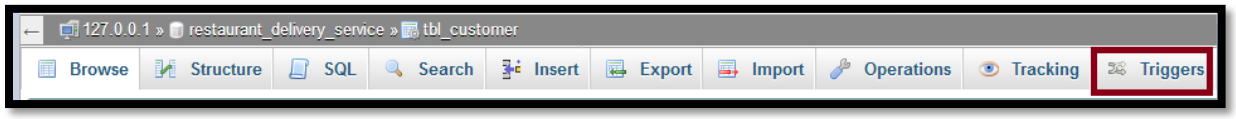


Figure: 1.38-Display of create trigger

Displays of Add Trigger View:

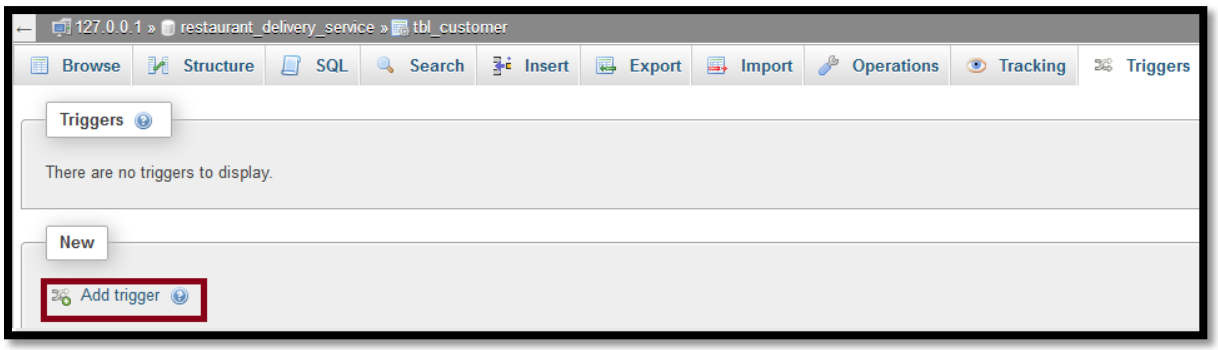


Figure: 1.39-Display of add trigger

Create View Code for Trigger:

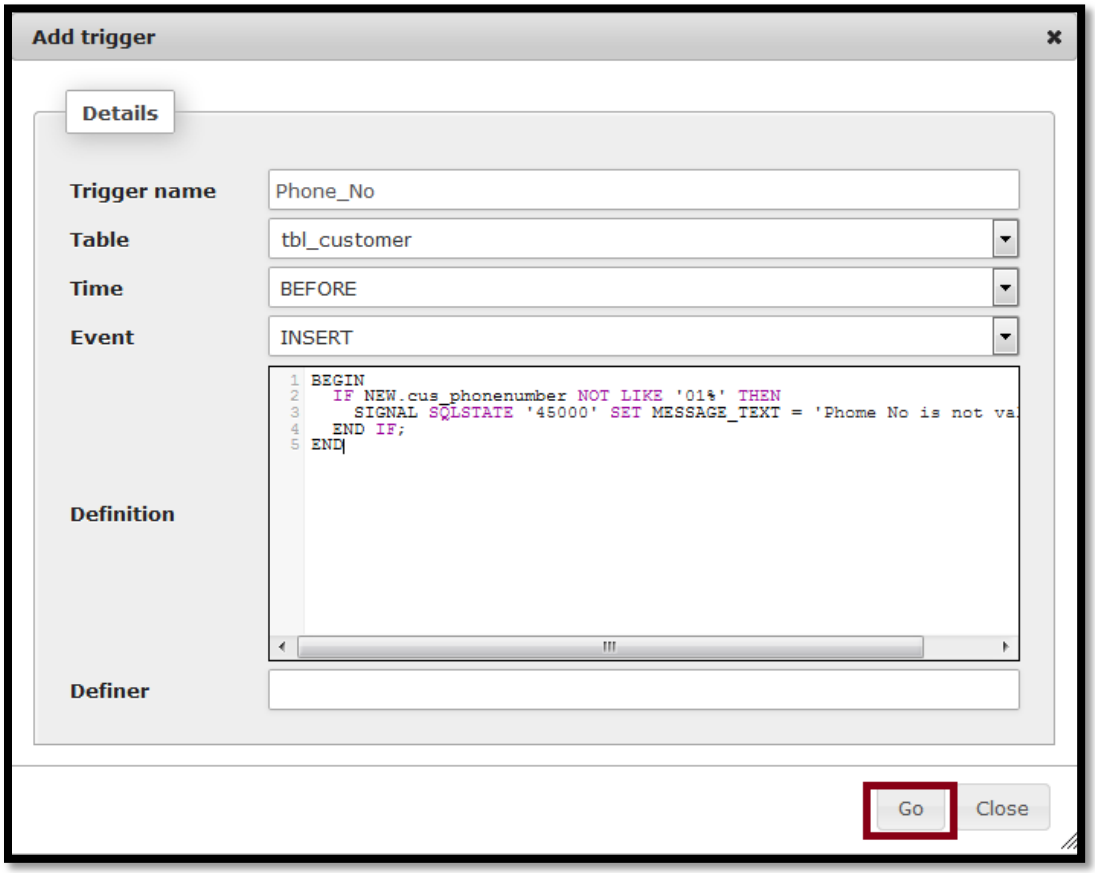


Figure: 1.40-Display of create trigger

Code for trigger:

```
BEGIN
  IF NEW.cus_phonenumber NOT LIKE '01%' THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Phome No is not valid';
  END IF;
END
```

Result: Trigger Phone_No has been created.

```
CREATE TRIGGER `Phone_No` BEFORE INSERT ON `tbl_customer`
FOR EACH
ROW BEGIN
IF NEW.cus_phonenumber NOT LIKE '01%'
THEN
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = 'Phome No is not valid';

END IF ;

END
```

Figure: 1.41- Display of "trigger" create successfully

Trigger Check by wrong value: Here ***cus_phonenumber*** value dose not maintained trigger rules.

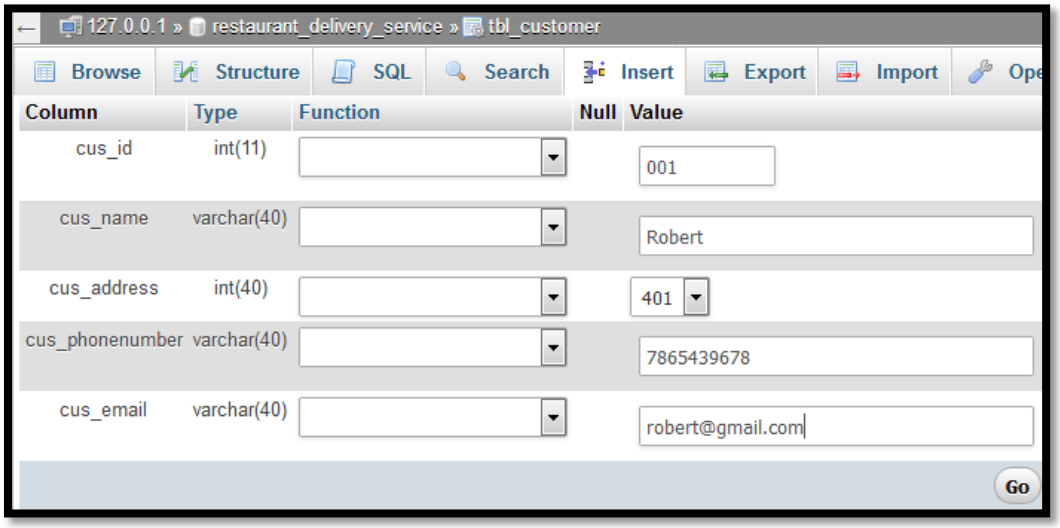


Figure: 1.42-Display of check trigger value

Output: Here showing ***Phone No is not valid.***

Figure: 1.43-Display of phone no is not valid

Right value check by "phone no": Here ***cus_phonenumber*** value maintained trigger rules.

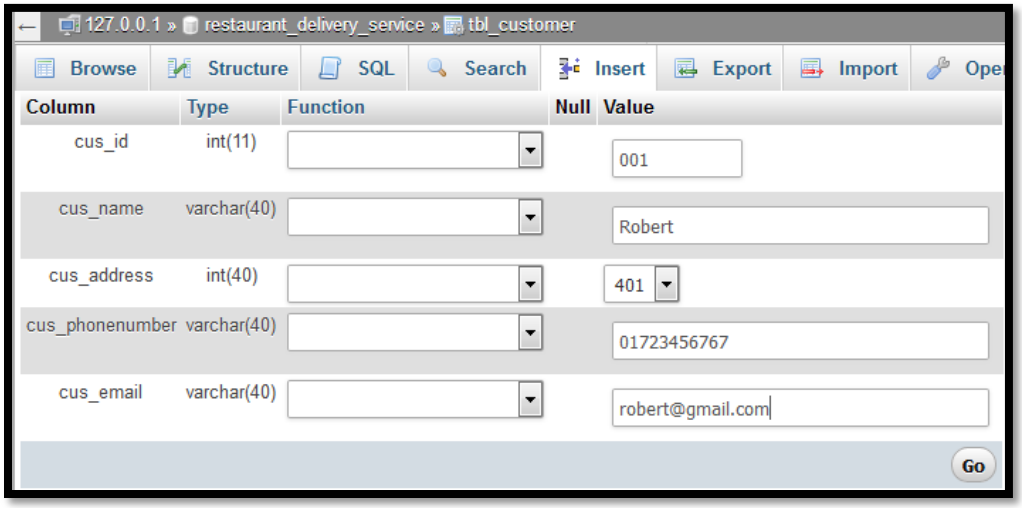


Figure: 1.44-Right value checking

Result: value insert successfully.

```
INSERT INTO `restaurant_delivery_service`.`tbl_customer` (
  `cus_id` ,
  `cus_name` ,
  `cus_address` ,
  `cus_phonenumber` ,
  `cus_email`
)
VALUES (
  '001', 'Robert', '401', '01723456767', 'robert@gmail.com'
);
```

Figure: 1.45- Display of " one row inserted" successfully

Output: After inserted ***cus_phonenumber*** value.

+ Options

cus_id	cus_name	cus_address	cus_phonenumber	cus_email
601	Robert	401	07911177233	robert@gmail.com
602	Ava	402	07911177211	ava@gmail.com
603	Amelia	403	07911177280	amelia@gmail.com
604	Bonny	404	07911177555	bonny@gmail.com
605	Emily	405	07934721000	emily@gmail.com
606	Olivia	406	07911177555	olivia@gmail.com
1	Robert	401	01723456767	robert@gmail.com

Figure: 1.46-Table for insert successfully

Evaluation:

Now a day ***restaurant*** is very famous place for every person. In our country ***there are many kind of restaurant*** which they provide testy food and delivery different food. There are many kinds of ***restaurant open in 24 hours***. Which people collect food all time by different price? So that I have selected this topic for this reasons. I have ***completed the entire major task*** in my selected topic. I will try to ***cover all the major features in my database***. But after developing this system ***I have realizes some other features should be need this system***. First of the entire customer can not ***send massage to employee for any problem*** and they do not get any ***notification for payment or delivery***. So it is current problem for database system. So it must be ***cover further development***. Otherwise I think this system designed very well. And it will be working very well in the real life.

Conclusion:

In this assignment I have ***gained different knowledge about database system***. Including I have gain ***SQL code, ERD, create table, query, sub-query, view table and trigger etc***. my experience will help me full better success in the near future where database problems will come out.

Self assessment:

	%	Fail	Insufficient	Pass	Good	Very Good	Excellent
ERD/Schema	10						✓
Implementation	10					✓	
Queries(Explanation And Execution)	40						✓
Trigger And View	20						✓
Self-Assessment Sheet And Evaluation	20					✓	
Total	100						

Bibliography

Anon., n.d. */en-us/sql/t-sql/statements/create-database-sql-server-transact-sql*. [Online] Available at: https://docs.microsoft.com [Accessed 7 November 2017].

Anon., n.d. */sql/sql_syntax.asp*. [Online] Available at: https://www.w3schools.com [Accessed 8 November 2017].

Anon., n.d. *sql*. [Online] Available at: https://www.w3schools.com/ [Accessed 9 November 2017].

Anon., n.d. *sql/sql_create_db.asp*. [Online] Available at: https://www.w3schools.com/ [Accessed 10 November 2017].