



TOPIC:

**A DESIGN AND IMPLEMENTATION OF A
RAILWAY INFORMATION SYSTEM**

Submitted to

*Basic Sciences and Humanities Department
Of*

Cyprus International University

Faculty of Engineering

Department Of Computer And Programming Engineering

**DATABASE MANAGEMENT SYSTEMS AND
PROGRAMMING I (CMP343)**

By

DJOUKAN TCHINDA WILLY STEVEN

22016432

&

JOSEPHINE YANKA-BABU JACOBS

22002653

Saturday 14th January, 2023.

TABLE OF CONTENT

TABLE OF CONTENT	1
1- INTRODUCTION	2
2- ENTITY RELATIONAL DIAGRAM (ERD)	3
3- RELATIONAL MODEL	4
4- DATA DEFINITION LANGUAGE (DDL)	5
5- DATA MANIPULATION LANGUAGE (DML).....	6
5.1 -- Database: `railway_info_sys`	8
6- NORMALIZATION.....	14
7- RETRIEVAL INFORMATION FOR MANAGEMENT	15

1- INTRODUCTION

The main aim of this report is to develop a system that provide facility for reporting relevant information about a railway system. Actually, through this system, we can retrieve data about the railway company like the number of passengers that did a reservation or bought a ticket on a specific date, the routes each train take on daily basis. We can also find the busy schedules, rush hours, most preferred routes the trains take daily, one of the station representative can track the details of the coaches working hours, his availability on a given date and time.

Furthermore, each table is related to one another which is represented on the entity relational diagram (ERD) shown on figure 1 and through the relational model we can easily see their associativity show on figure 2.

The database was created by the Data Defintion Language (DDL) which was named “**railway_info_sys**” and we recorded the data through the Data Manipulation Language (DML) for each table respectively.

The normalization of the tables on the other hand, was been done through the 1NF, 2NF and 3NF. This is to find out any anomaly in the database to prevent mostly redundancy and waste of space. By using this technique, it organizes data into groups to form stable, flexible and adaptive entities for the company which enhances their efficiency.

This system is develop to ease the management of the railway company, to be able to trace or retrieve every details when need in the near future either for update, modify, alter, insert or delete any detail in the database.

2- ENTITY RELATIONAL DIAGRAM (ERD)

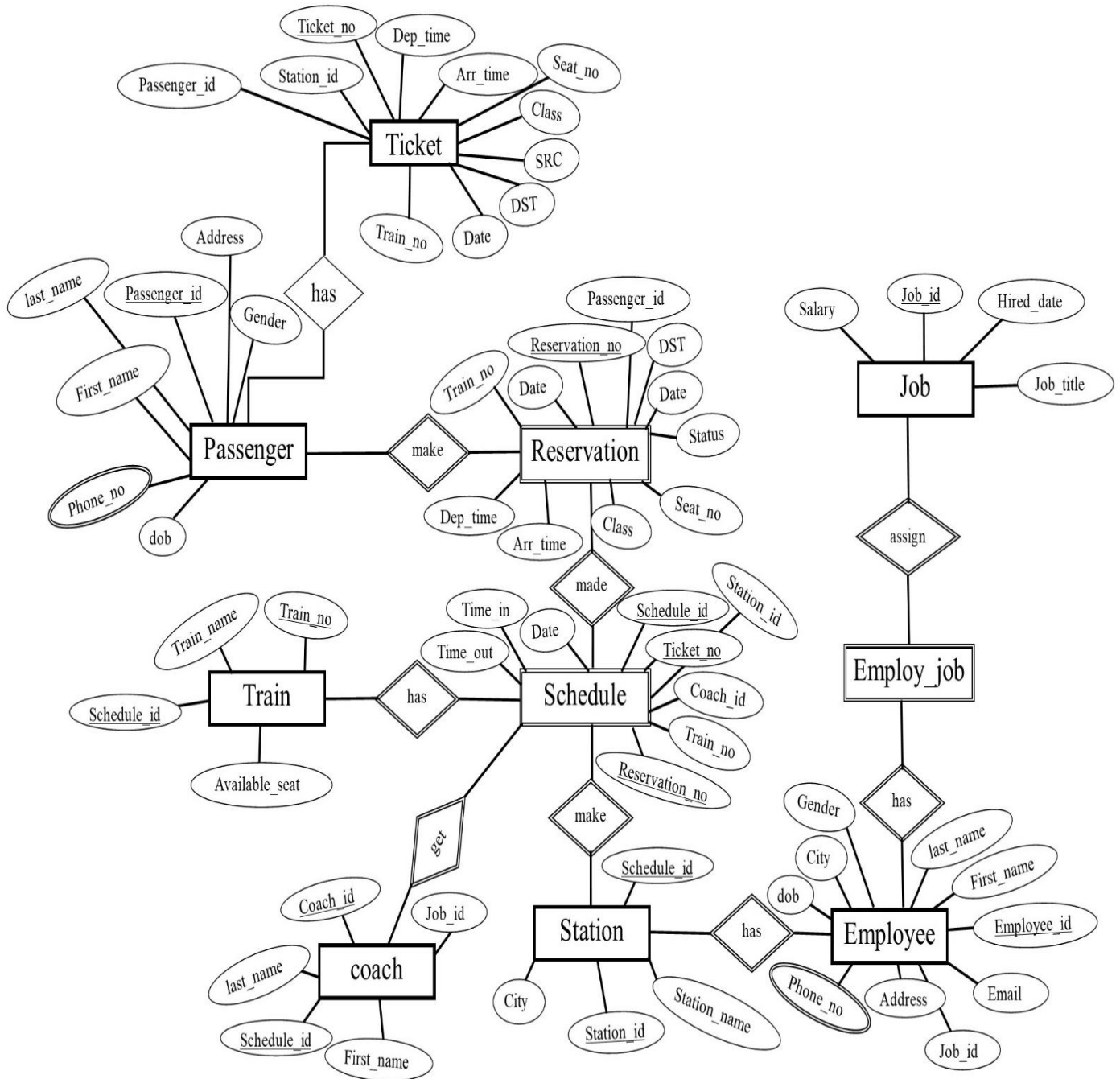


Figure 1: ER diagram

3- RELATIONAL MODEL

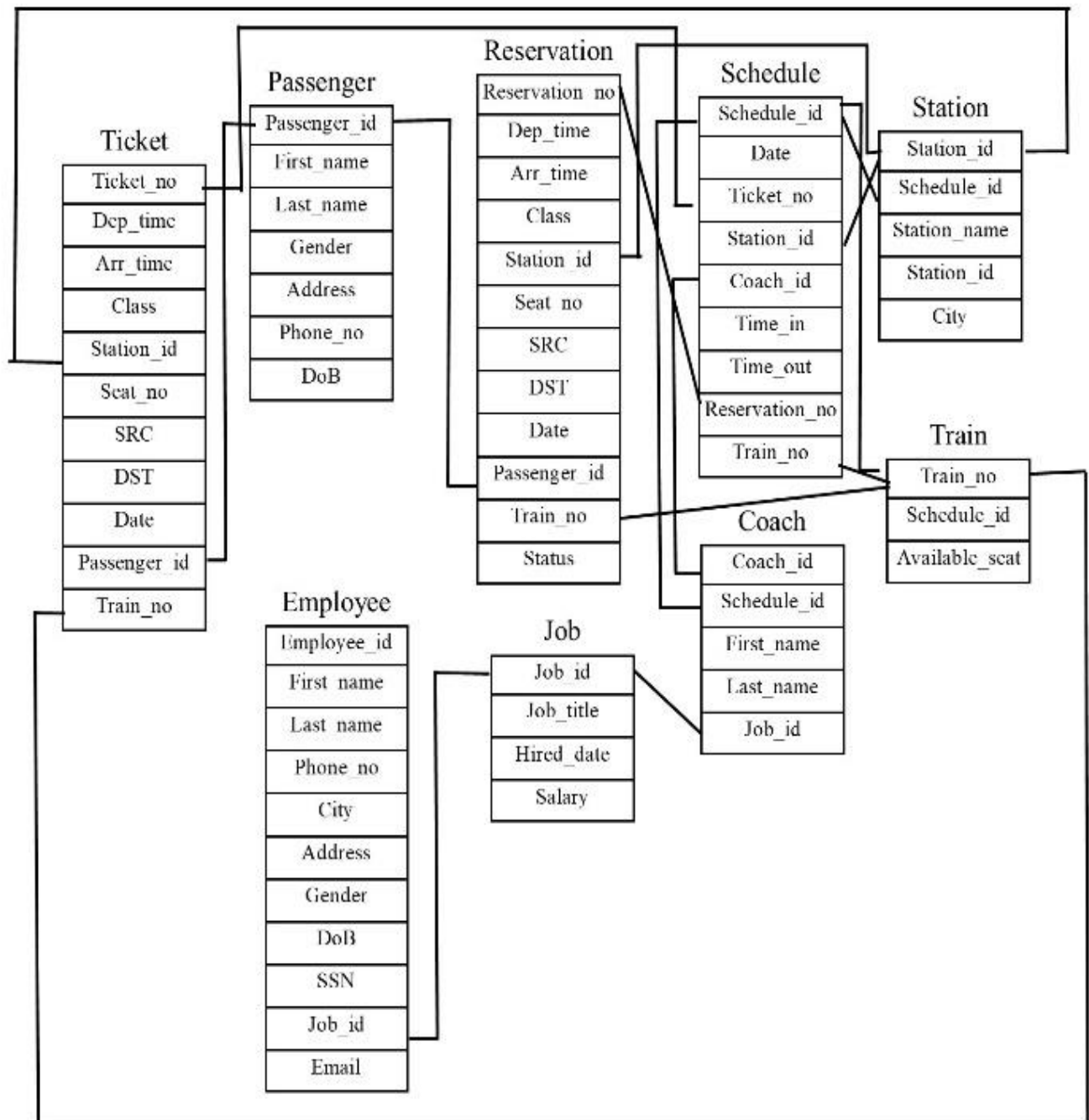
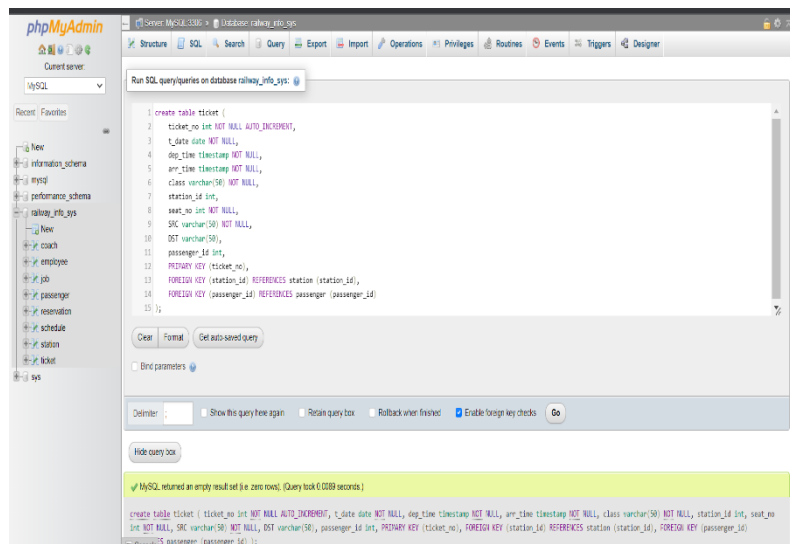
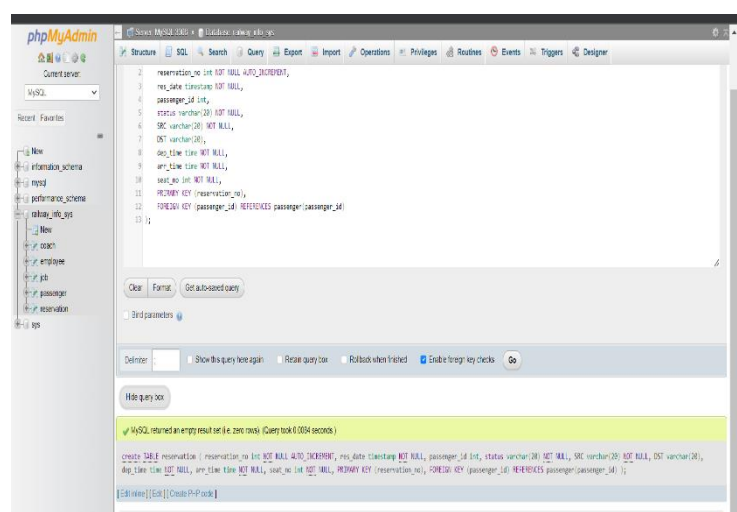
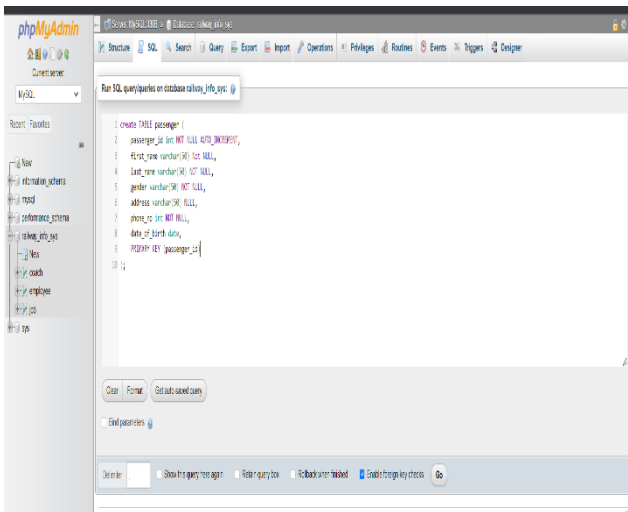
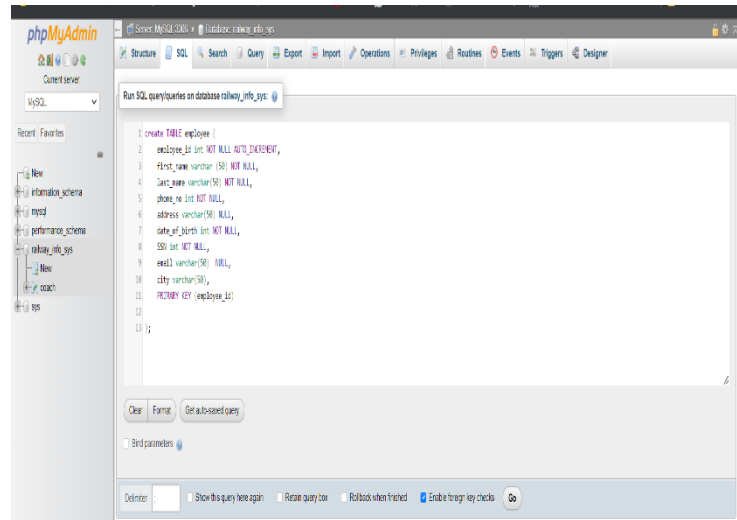
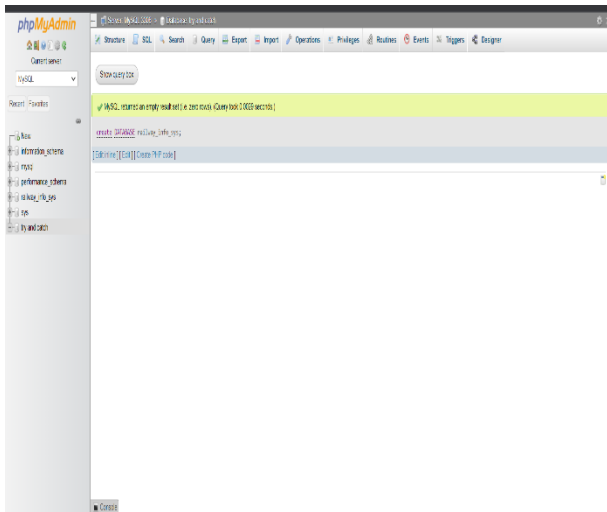


Figure 2: Relational Model

4- DATA DEFINITION LANGUAGE (DDL)

Creation of Database and few table schemas;



5- DATA MANIPULATION LANGUAGE (DML)

This section shows the different kind of data manipulation in the database;

The screenshot shows the phpMyAdmin interface with the 'passenger' table selected. The SQL query editor contains an INSERT statement. The table structure is shown on the right.

```
1 INSERT INTO `passenger` (`passenger_id`, `first_name`, `last_name`,  
2 `gender`, `address`, `phone_no`, `date_of_birth`)  
3 VALUES ('1', 'will', 'tchinda', 'male', 'haspolat', '45744545', '1996-02-06'),  
4 ('', 'steve', 'djoukan', 'male', 'ortakoy', '4577841', '1996-10-06'),  
5 ('', 'kopel', 'valdo', 'male', 'girne', '788878', '1994-02-05'),  
6 ('', 'jojo', 'jacobs', 'female', 'lefke', '44447455', '2002-01-02'),  
7 ('', 'lavalchini', 'zara', 'male', 'ohoo', '4511241', '2005-12-26'),  
8 ('', 'patricia', 'tchinda', 'female', 'nkong', '7844515', '1963-05-22'),  
9 ('', 'calota', 'kagho', 'female', 'douala', '45445454', '1998-06-10'),  
10 ('', 'felixa', 'tchinda', 'female', 'yaounde', '41124152', '2001-12-12')
```

passenger_id	first_name	last_name	gender	address	phone_no	date_of_birth
1	will	tchinda	male	haspolat	45744545	1996-02-06

The screenshot shows the phpMyAdmin interface with the 'reservation' table selected. The SQL query editor contains an INSERT statement. The table structure is shown on the right.

```
1 INSERT INTO `reservation` (`reservation_no`, `res_date`, `passenger_id`, `status`, `SRC`, `DST`, `dep_time`,  
2 `arr_time`, `seat_no`, `class`, `train_no`, `station_id`)  
3 VALUES ('1', '2023-01-10',  
4 '20:30:00', '17', 'approved', 'Nkong', 'yaounde', '21:00:00', '03:00:00', '15', '1AC', '36', '2181'),  
5 ('', '2023-01-10 20:35:00', '14', 'approved', 'douala', 'yaounde', '21:00:00', '24:00:00', '28', '1C', '53', '2123'),  
6 ('', '2023-01-10 20:40:00', '12', 'approved', 'Nkong', 'douala', '21:00:00', '23:00:00', '18', '1C', '26', '2181'),  
7 ('', '2023-01-10 20:45:00', '15', 'approved', 'loua', 'limbe', '21:00:00', '21:45:00', '14', '1AC', '25', '2181'),  
8 ('', '2023-01-10 20:50:00', '16', 'approved', 'ligoa', 'bare', '21:00:00', '22:30:00', '9', '1C', '27', '2181'),  
9 ('', '2023-01-10 20:55:00', '18', 'approved', 'loua', 'mbanga', '21:00:00', '22:00:00', '17', '1C', '36', '2181')
```

reservation_no	res_date	passenger_id	status	SRC	DST	dep_time	arr_time	seat_no	class	train_no	station_id
1	2023-01-10	17	approved	Nkong	yaounde	21:00:00	03:00:00	15	1AC	36	2181

The screenshot shows the phpMyAdmin interface with the 'passenger' table selected. The SQL query editor contains a DELETE statement. The table structure is shown on the right.

```
1 DELETE FROM `passenger`  
2 WHERE address in ('haspolat', 'girne', 'douala')
```

passenger_id	first_name	last_name	gender	address	phone_no	date_of_birth
1	will	djoukan	male	downtown	1245879	1996-02-06

The screenshot shows the phpMyAdmin interface with the 'passenger' table selected. The SQL query editor contains a SELECT statement. The table structure is shown on the right.

```
1 SELECT * FROM `passenger`;
```

passenger_id	first_name	last_name	gender	address	phone_no	date_of_birth
1	will	djoukan	male	downtown	1245879	1996-02-06
2	alanis	tohinda	female	uptown hehe	8963523	2000-05-21
3	kopel	djonkou	male	ohio	4512639	1994-02-06
4	corine	kengmo	female	utagae	7856412	1991-06-10
6	pope	chambelain	male		5478963	1965-02-25
7	jery	elliot	male	spring street	4574896	1996-02-18
8	quavo	bale	male		4789563	2000-01-22
9	riley	denovan	female		4523698	1998-12-25

phpMyAdmin - Server: MySQL3306 - Database: railway_info_sys - Table: reservation

Current server: MySQL

Showing rows 0 - 7 (8 total. Query took 0.0010 seconds.)

SELECT * FROM 'reservation' WHERE DST IN (SELECT DST FROM ticket);

Extra options

	reservation_no	res_date	passenger_id	status	SRC	DST	dep_time	arr_time	seat_no	class	train_no	station_id
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	2023-01-11 18:55:49	2	approved	ohio	downtown	12:00:00	13:30:00	1	EC	25	2001
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	2023-01-11 18:55:49	4	approved	ohio	downtown	12:00:00	13:30:00	2	1AC	25	2125
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	2023-01-11 18:55:49	3	approved	downtown	ohio	12:00:00	13:30:00	3	FC	25	2125
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	4	2023-01-11 18:55:49	6	approved	lavalite hills	downtown	12:00:00	13:30:00	4	EC	25	2114
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	5	2023-01-11 18:55:49	5	cancel	beverlyhill	ohio	14:00:00	16:30:00	1	FC	52	2101
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	6	2023-01-10 20:30:00	17	approved	Nkong	yaounde	21:00:00	03:00:00	15	2AC	26	2101
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	7	2023-01-10 20:35:00	14	approved	douala	yaounde	21:00:00	24:00:00	20	FC	52	2125
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	8	2023-01-10 20:40:00	12	approved	Nkong	douala	21:00:00	23:00:00	10	FC	26	2101

Check all With selected: Edit Copy Delete Export

phpMyAdmin - Server: MySQL3306 - Database: railway_info_sys - Table: coach

Current server: MySQL

Run SQL query/queries on table railway_info_sys.coach:

```

1 INSERT INTO 'coach' ('coach_id', 'first_name', 'last_name', 'schedule_id', 'job_id')
2 VALUES ('1','bryan','kobe','5','5'),
3 ('','wilfrid','stephen','4','5'),
4 ('','alex','nathan','3','5'),
5 ('','ryan','schwender','1','5'),
6 ('','donald','mark','4','5'),

```

coach_id
first_name
last_name
schedule_id
job_id

SELECT * SELECT INSERT UPDATE DELETE Clear Format Get auto-saved query

Bind parameters

phpMyAdmin - Server: MySQL3306 - Database: railway_info_sys - Table: employee

Current server: MySQL

Run SQL query/queries on table railway_info_sys.employee:

```

1 UPDATE 'employee' SET 'gender'='female'
2 WHERE first_name = 'sasha' OR first_name = 'gwen' OR first_name = 'nadesh' OR first_name = 'eleonore';

```

employee_id
first_name
last_name
phone_no
address
date_of_birth
SSN
email
city
job_id
gender

SELECT * SELECT INSERT UPDATE DELETE Clear Format Get auto-saved query

Bind parameters

Delimiter ; Show this query here again Retain query box Rollback when finished Enable foreign key checks Simulate query Go

5.1 -- Database: `railway_info_sys`

-- Table structure for table `coach`

```
CREATE TABLE IF NOT EXISTS `coach` (  
  `coach_id` int NOT NULL AUTO_INCREMENT,  
  `first_name` varchar(50) NOT NULL,  
  `last_name` varchar(50) NOT NULL,  
  `schedule_id` int DEFAULT NULL,  
  `job_id` int NOT NULL,  
  PRIMARY KEY (`coach_id`),  
  KEY `job_id` (`job_id`),  
  KEY `schedule_id` (`schedule_id`)  
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=utf8mb4  
COLLATE=utf8mb4_0900_ai_ci;
```

```
INSERT INTO `coach` (`coach_id`, `first_name`, `last_name`, `schedule_id`, `job_id`) VALUES  
(1, 'john', 'kaminga', 2, 5),  
(2, 'ricky', 'dave', 1, 5),  
(3, 'rigobert', 'josh', 4, 5),  
(4, 'gerald', 'peter', 3, 5);
```

-- Constraints for table `coach`

```
ALTER TABLE `coach`  
  ADD CONSTRAINT `coach_ibfk_1` FOREIGN KEY (`job_id`) REFERENCES `job` (`job_id`) ON  
DELETE RESTRICT ON UPDATE RESTRICT,  
  ADD CONSTRAINT `coach_ibfk_2` FOREIGN KEY (`schedule_id`) REFERENCES `schedule`  
(`schedule_id`) ON DELETE RESTRICT ON UPDATE RESTRICT;  
COMMIT;
```

-- Table structure for table `employee`

```
CREATE TABLE IF NOT EXISTS `employee` (  
  `employee_id` int NOT NULL AUTO_INCREMENT,  
  `first_name` varchar(50) NOT NULL DEFAULT 'NOT NULL',  
  `last_name` varchar(50) NOT NULL DEFAULT 'NOT NULL',  
  `phone_no` int NOT NULL,  
  `date_of_birth` date NOT NULL,  
  `gender` varchar(10) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL  
DEFAULT 'NOT NULL',  
  `SSN` int NOT NULL,  
  `email` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci DEFAULT NULL,  
  `city` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci DEFAULT NULL,  
  `address` varchar(50) DEFAULT NULL,  
  `job_id` int DEFAULT NULL,  
  PRIMARY KEY (`employee_id`),  
  KEY `employee_ibfk_1` (`job_id`)  
) ENGINE=InnoDB AUTO_INCREMENT=13 DEFAULT CHARSET=utf8mb4  
COLLATE=utf8mb4_0900_ai_ci;
```

```

INSERT INTO `employee` (`employee_id`, `first_name`, `last_name`, `phone_no`, `date_of_birth`,
`gender`, `SSN`, `email`, `city`, `address`, `job_id`) VALUES
(1, 'sasha', 'maeve', 2564574, '1963-10-20', 'female', 784, 'que@gmailcom', 'ohio', 'dalington street', 2),
(2, 'gwen', 'gwei', 45477454, '1992-02-06', 'female', 546, '', 'douala', 'cite verte', 8),
(3, 'calid', 'khal', 4789562, '1975-06-20', 'male', 545, 'bew@gmailcom', 'beverlyhill', 'hillhigh street', 1),
(4, 'caleb', 'antoine', 1254789, '1985-05-12', 'male', 778, 'ash@gmailcom', 'orgen', 'hallway street', 4),
(5, 'nadesh', 'wiley', 548966, '1988-12-12', 'female', 565, 'wal@gmailcom', 'guatemala', 'queens street', 3),
(6, 'rosine', 'bouda', 45122145, '1989-12-11', 'female', 112, '', 'douala', 'toiture rouge', 7),
(7, 'eleonore', 'tchang', 745445, '1973-11-25', 'female', 331, '', 'nkong', 'rond point', 6);

```

-- Constraints for table `employee`

```

ALTER TABLE `employee`
ADD CONSTRAINT `employee_ibfk_1` FOREIGN KEY (`job_id`) REFERENCES `job` (`job_id`)
ON DELETE CASCADE ON UPDATE CASCADE;
COMMIT;

```

-- Table structure for table `job`

```

CREATE TABLE IF NOT EXISTS `job` (
`job_id` int NOT NULL AUTO_INCREMENT,
`job_title` varchar(50) NOT NULL,
`hired_date` date NOT NULL,
`salary` int NOT NULL,
PRIMARY KEY (`job_id`)
) ENGINE=InnoDB AUTO_INCREMENT=9 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;

```

```

INSERT INTO `job` (`job_id`, `job_title`, `hired_date`, `salary`) VALUES
(1, 'ticket controler', '2012-10-12', 70000),
(2, 'cashier', '2018-01-20', 80000),
(3, 'head officer', '2010-05-02', 350000),
(4, 'station agent', '2015-08-15', 75000),
(5, 'coach', '2010-11-30', 85000),
(6, 'station mistress', '2001-06-25', 500000),
(7, 'accountant', '2005-10-05', 250000),
(8, 'receptionist', '2010-01-15', 50000);
COMMIT;

```

-- Table structure for table `passenger`

```

CREATE TABLE IF NOT EXISTS `passenger` (
`passenger_id` int NOT NULL AUTO_INCREMENT,
`first_name` varchar(50) NOT NULL DEFAULT 'NOT NULL',
`last_name` varchar(50) NOT NULL DEFAULT 'NOT NULL',
`date_of_birth` date DEFAULT NULL,
`gender` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
`phone_no` int NOT NULL,
`address` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL
DEFAULT 'NOT NULL',

```

```

PRIMARY KEY (`passenger_id`)
) ENGINE=InnoDB AUTO_INCREMENT=18 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;

```

```

INSERT INTO `passenger` (`passenger_id`, `first_name`, `last_name`, `date_of_birth`, `gender`,
`phone_no`, `address`) VALUES
(1, 'will', 'djoukan', '1996-02-06', 'male', 1245879, 'downtown'),
(2, 'alanis', 'tchinda', '2000-05-21', 'female', 8963523, 'uptown hehe'),
(3, 'kopel', 'djoukou', '1994-02-06', 'male', 4512639, 'ohio'),
(4, 'corine', 'kengmo', '1991-06-10', 'female', 7856412, 'utagae'),
(5, 'pope', 'chambelain', '1965-02-25', 'male', 5478963, 'guatemala'),
(6, 'jerry', 'elliott', '1996-02-18', 'male', 4574896, 'spring street'),
(7, 'quavo', 'bale', '2000-01-22', 'male', 4789563, 'beverlyhill'),
(8, 'riley', 'denovan', '1998-12-25', 'female', 4523698, 'orgen'),
(9, 'steve', 'djoukan', '1996-10-06', 'male', 4577841, 'ortakoy'),
(10, 'lavalbhiny', 'zara', '2005-12-26', 'male', 4511241, 'ohoo'),
(11, 'patricia', 'tchinda', '1963-05-22', 'female', 7844515, 'nkong'),
(12, 'calota', 'kagho', '1998-06-10', 'female', 45445454, 'douala'),
(13, 'felixa', 'tchinda', '2001-12-12', 'female', 41124152, 'yaounde');
COMMIT;

```

-- Table structure for table `reservation`

```

CREATE TABLE IF NOT EXISTS `reservation` (
  `reservation_no` int NOT NULL AUTO_INCREMENT,
  `res_date` timestamp NOT NULL,
  `passenger_id` int NOT NULL,
  `status` varchar(20) NOT NULL,
  `SRC` varchar(20) NOT NULL,
  `DST` varchar(20) DEFAULT NULL,
  `dep_time` time NOT NULL,
  `arr_time` time NOT NULL,
  `seat_no` int NOT NULL,
  `class` varchar(50) DEFAULT NULL,
  `train_no` int DEFAULT NULL,
  `station_id` int NOT NULL,
  PRIMARY KEY (`reservation_no`),
  KEY `station_id` (`station_id`),
  KEY `train_no` (`train_no`),
  KEY `passenger_id` (`passenger_id`)
) ENGINE=InnoDB AUTO_INCREMENT=12 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;

```

```

INSERT INTO `reservation` (`reservation_no`, `res_date`, `passenger_id`, `status`, `SRC`, `DST`,
`dep_time`, `arr_time`, `seat_no`, `class`, `train_no`, `station_id`) VALUES
(1, '2023-01-11 16:55:49', 2, 'approved', 'ohio', 'downtown', '12:00:00', '13:30:00', 1, 'EC', 25, 2001),
(2, '2023-01-11 16:55:49', 4, 'approved', 'ohio', 'downtown', '12:00:00', '13:30:00', 2, '1AC', 25, 2125),
(3, '2023-01-11 16:55:49', 3, 'approved', 'downtown', 'ohio', '12:00:00', '13:30:00', 3, 'FC', 25, 2125),
(4, '2023-01-11 16:55:49', 6, 'approved', 'lavalte hills', 'downtown', '12:00:00', '13:30:00', 4, 'EC', 25,
2114),
(5, '2023-01-11 16:55:49', 5, 'cancel', 'beverlyhill', 'ohio', '14:00:00', '15:30:00', 1, 'FC', 52, 2101),

```

```
(6, '2023-01-10 18:30:00', 11, 'approved', 'Nkong', 'yaounde', '21:00:00', '03:00:00', 15, '2AC', 26, 2101),
(7, '2023-01-10 18:35:00', 8, 'approved', 'douala', 'yaounde', '21:00:00', '24:00:00', 20, 'FC', 52, 2125),
(8, '2023-01-10 18:40:00', 12, 'approved', 'Nkong', 'douala', '21:00:00', '23:00:00', 10, 'FC', 26, 2101),
(9, '2023-01-10 18:45:00', 7, 'approved', 'buea', 'limbe', '21:00:00', '21:45:00', 14, '1AC', 25, 2101),
(10, '2023-01-10 18:50:00', 1, 'approved', 'Ngoa', 'bare', '21:00:00', '22:30:00', 9, 'EC', 27, 2101),
(11, '2023-01-10 18:55:00', 10, 'approved', 'Ioum', 'mbanga', '21:00:00', '22:00:00', 7, 'FC', 26, 2101);
```

```
-- Constraints for table `reservation`
```

```
ALTER TABLE `reservation`
  ADD CONSTRAINT `reservation_ibfk_1` FOREIGN KEY (`station_id`) REFERENCES `station`
  (`station_id`) ON DELETE RESTRICT ON UPDATE RESTRICT,
  ADD CONSTRAINT `reservation_ibfk_2` FOREIGN KEY (`train_no`) REFERENCES `train`
  (`train_no`) ON DELETE RESTRICT ON UPDATE RESTRICT,
  ADD CONSTRAINT `reservation_ibfk_3` FOREIGN KEY (`passenger_id`) REFERENCES
  `passenger` (`passenger_id`) ON DELETE RESTRICT ON UPDATE RESTRICT;
COMMIT;
```

```
-- Table structure for table `schedule`
```

```
CREATE TABLE IF NOT EXISTS `schedule` (
  `schedule_id` int NOT NULL AUTO_INCREMENT,
  `time_in` time NOT NULL,
  `time_out` time NOT NULL,
  `coach_id` int DEFAULT NULL,
  `reservation_id` int DEFAULT NULL,
  `ticket_no` int DEFAULT NULL,
  `train_no` int DEFAULT NULL,
  `station_id` int DEFAULT NULL,
  PRIMARY KEY (`schedule_id`),
  KEY `coach_id` (`coach_id`),
  KEY `reservation_id` (`reservation_id`),
  KEY `ticket_no` (`ticket_no`),
  KEY `train_no` (`train_no`),
  KEY `station_id` (`station_id`)
) ENGINE=InnoDB AUTO_INCREMENT=6 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;
```

```
INSERT INTO `schedule` (`schedule_id`, `time_in`, `time_out`, `coach_id`, `reservation_id`, `ticket_no`,
`train_no`, `station_id`) VALUES
(1, '15:50:00', '16:00:00', 3, 5, 1001, 25, 2001),
(2, '13:30:00', '12:00:00', 1, 2, 1005, 26, 2114),
(3, '15:30:00', '14:00:00', 1, 1, 1003, 52, 2114),
(4, '15:50:00', '14:50:00', 3, 3, 1004, 25, 2125),
(5, '13:30:00', '12:00:00', 3, 4, 1002, 25, 2101);
```

```
-- Constraints for table `schedule`
```

```
ALTER TABLE `schedule`
  ADD CONSTRAINT `schedule_ibfk_1` FOREIGN KEY (`coach_id`) REFERENCES `coach`
  (`coach_id`) ON DELETE RESTRICT ON UPDATE RESTRICT,
```

```

    ADD CONSTRAINT `schedule_ibfk_2` FOREIGN KEY (`reservation_id`) REFERENCES
`reservation` (`reservation_no`) ON DELETE RESTRICT ON UPDATE RESTRICT,
    ADD CONSTRAINT `schedule_ibfk_3` FOREIGN KEY (`ticket_no`) REFERENCES `ticket`
(`ticket_no`) ON DELETE RESTRICT ON UPDATE RESTRICT,
    ADD CONSTRAINT `schedule_ibfk_4` FOREIGN KEY (`train_no`) REFERENCES `train`
(`train_no`) ON DELETE RESTRICT ON UPDATE RESTRICT,
    ADD CONSTRAINT `schedule_ibfk_5` FOREIGN KEY (`station_id`) REFERENCES `station`
(`station_id`) ON DELETE RESTRICT ON UPDATE RESTRICT;
COMMIT;

```

-- Table structure for table `station`

```

CREATE TABLE IF NOT EXISTS `station` (
  `station_id` int NOT NULL,
  `station_name` varchar(50) NOT NULL,
  `city` varchar(50) NOT NULL,
  `schedule_id` int DEFAULT NULL,
  PRIMARY KEY (`station_id`),
  KEY `schedule_id` (`schedule_id`),
  KEY `station_id` (`station_id`),
  KEY `station_id_2` (`station_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

```

```

INSERT INTO `station` (`station_id`, `station_name`, `city`, `schedule_id`) VALUES
(2001, 'Westbrone', 'ohio', 5),
(2101, 'Bronx', 'downtown', 1),
(2114, 'Trinity', 'beverlyhill', 2),
(2120, 'Dracant', 'treyo', 3),
(2125, 'downside', 'califort', 5),
(2126, 'Stark', 'downtown', 4);

```

-- Constraints for table `station`

```

ALTER TABLE `station`
  ADD CONSTRAINT `station_ibfk_1` FOREIGN KEY (`schedule_id`) REFERENCES `schedule`
(`schedule_id`) ON DELETE RESTRICT ON UPDATE RESTRICT;
COMMIT;

```

-- Table structure for table `ticket`

```

CREATE TABLE IF NOT EXISTS `ticket` (
  `ticket_no` int NOT NULL AUTO_INCREMENT,
  `t_date` datetime NOT NULL,
  `dep_time` time NOT NULL,
  `arr_time` time NOT NULL,
  `class` varchar(50) NOT NULL,
  `station_id` int DEFAULT NULL,
  `seat_no` int NOT NULL,
  `SRC` varchar(50) NOT NULL,
  `DST` varchar(50) DEFAULT NULL,
  `passenger_id` int DEFAULT NULL,

```

```

`train_no` int DEFAULT NULL,
PRIMARY KEY (`ticket_no`),
KEY `passenger_id` (`passenger_id`),
KEY `station_id` (`station_id`),
KEY `train_no` (`train_no`)
) ENGINE=InnoDB AUTO_INCREMENT=1006 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;

INSERT INTO `ticket` (`ticket_no`, `t_date`, `dep_time`, `arr_time`, `class`, `station_id`, `seat_no`,
`SRC`, `DST`, `passenger_id`, `train_no`) VALUES
(1001, '2023-01-11 14:00:00', '14:50:00', '15:50:48', 'EC', 2001, 8, 'oweho', 'ohio', 8, 26),
(1002, '2023-01-11 14:05:00', '14:50:00', '16:20:00', 'EC', 2120, 9, 'treoyo', 'califort', 7, 25),
(1003, '2023-01-11 14:30:00', '14:50:00', '15:30:00', '2AC', 2101, 7, 'tradot', 'downtown', 1, 25),
(1004, '2023-01-11 14:40:00', '14:50:00', '16:20:00', 'FC', 2125, 1, 'treoyo', 'yaounde', 9, 25),
(1005, '2023-01-11 14:40:00', '16:50:00', '17:20:00', 'FC', 2120, 6, 'queens', 'douala', 10, 26);

-- Constraints for table `ticket`

ALTER TABLE `ticket`
  ADD CONSTRAINT `ticket_ibfk_1` FOREIGN KEY (`passenger_id`) REFERENCES `passenger`
(`passenger_id`) ON DELETE RESTRICT ON UPDATE RESTRICT,
  ADD CONSTRAINT `ticket_ibfk_2` FOREIGN KEY (`station_id`) REFERENCES `station`
(`station_id`) ON DELETE RESTRICT ON UPDATE RESTRICT,
  ADD CONSTRAINT `ticket_ibfk_3` FOREIGN KEY (`train_no`) REFERENCES `train` (`train_no`)
ON DELETE RESTRICT ON UPDATE RESTRICT;
COMMIT;

-- Table structure for table `train`

DROP TABLE IF EXISTS `train`;
CREATE TABLE IF NOT EXISTS `train` (
  `train_no` int NOT NULL,
  `available_seat` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT
NULL,
  `schedule_id` int DEFAULT NULL,
  PRIMARY KEY (`train_no`),
  KEY `schedule_id` (`schedule_id`),
  KEY `train_no` (`train_no`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

INSERT INTO `train` (`train_no`, `available_seat`, `schedule_id`) VALUES
(25, '5', 1),
(26, '0', 2),
(27, '25', 4),
(52, '2', 3);

-- Constraints for table `train`
ALTER TABLE `train`
  ADD CONSTRAINT `train_ibfk_1` FOREIGN KEY (`schedule_id`) REFERENCES `schedule`
(`schedule_id`) ON DELETE RESTRICT ON UPDATE RESTRICT;
COMMIT;

```

6- NORMALIZATION

The database was been evaluated through the normalization technique to find out any redundancy while inserting, updating and deleting data in the database. We analyzed the data through the most common normal forms used. They are;

1- First Normal Form (1NF):

In here, all our cells contained exactly one value for each attribute define in the database.

2- Second Normal Form (2NF):

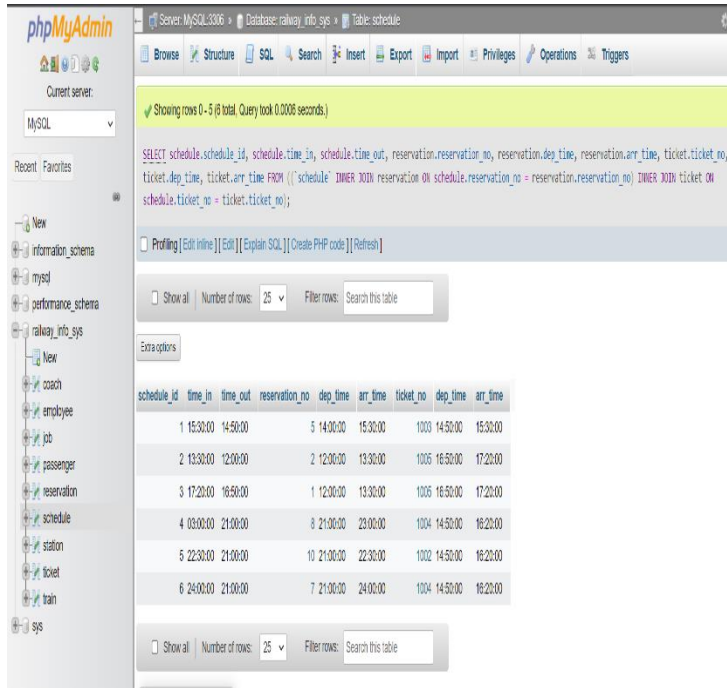
In here, we didn't have any partial dependency since every non-key column is dependent upon its **PRIMAY KEY**.

3- Third Normal Form (3NF):

In here, we didn't encountered any transitive dependency in the tables.

7- RETRIEVAL INFORMATION FOR MANAGEMENT

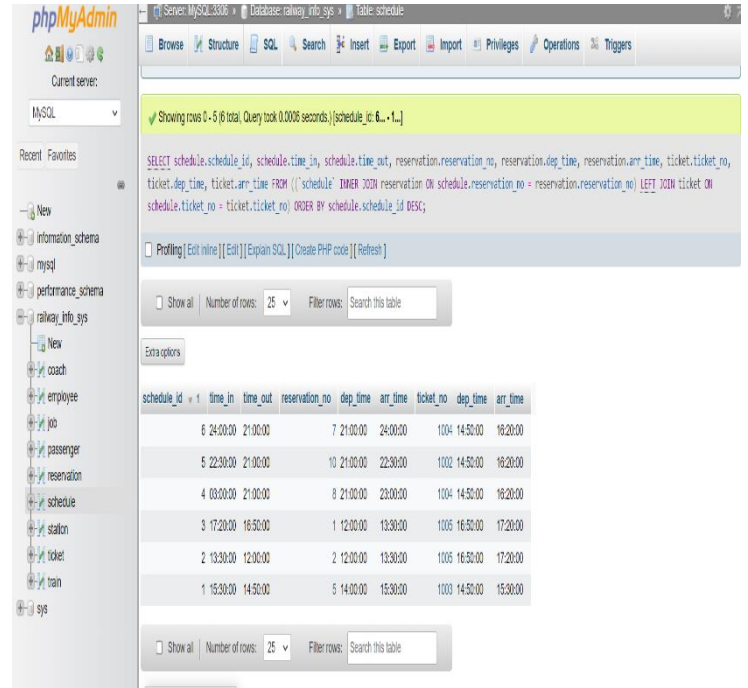
1. The query below shows the time a train passes through a station based on a time in and time out schedule. In this query, I used inner join, left join Group By DESC;



Showing rows 0 - 5 (6 total, Query took 0.0006 seconds.)

```
SELECT schedule.schedule_id, schedule.time_in, schedule.time_out, reservation.reservation_no, reservation.dep_time, reservation.arr_time, ticket.ticket_no, ticket.dep_time, ticket.arr_time FROM ((schedule INNER JOIN reservation ON schedule.reservation_no = reservation.reservation_no) INNER JOIN ticket ON schedule.ticket_no = ticket.ticket_no);
```

schedule_id	time_in	time_out	reservation_no	dep_time	arr_time	ticket_no	dep_time	arr_time
1	15:30:00	14:50:00	5	14:00:00	15:30:00	1000	14:50:00	15:30:00
2	13:30:00	12:00:00	2	12:00:00	13:30:00	1005	16:50:00	17:20:00
3	17:20:00	16:50:00	1	12:00:00	13:30:00	1006	16:50:00	17:20:00
4	03:00:00	21:00:00	8	21:00:00	23:00:00	1004	14:50:00	16:20:00
5	22:30:00	21:00:00	10	21:00:00	22:30:00	1002	14:50:00	16:20:00
6	24:00:00	21:00:00	7	21:00:00	24:00:00	1004	14:50:00	16:20:00

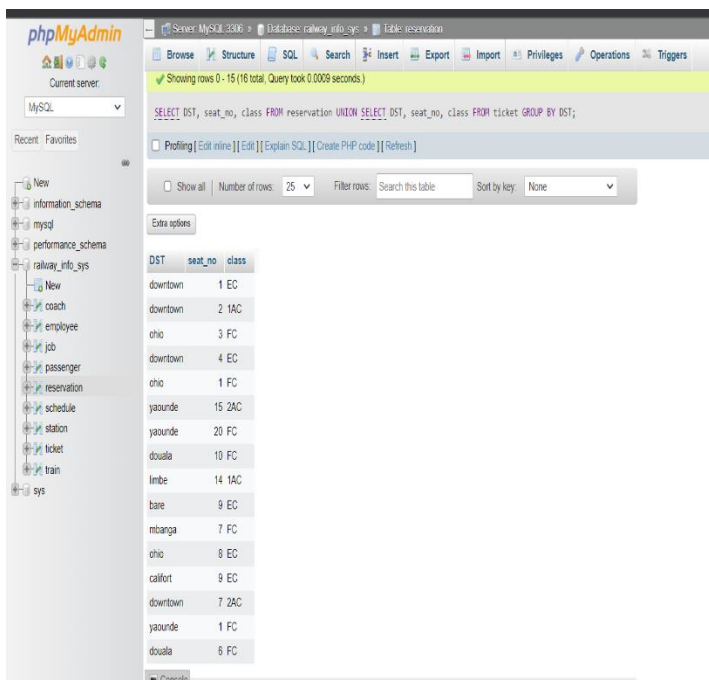


Showing rows 0 - 5 (6 total, Query took 0.0006 seconds.) [schedule_id: 6...]

```
SELECT schedule.schedule_id, schedule.time_in, schedule.time_out, reservation.reservation_no, reservation.dep_time, reservation.arr_time, ticket.ticket_no, ticket.dep_time, ticket.arr_time FROM ((schedule INNER JOIN reservation ON schedule.reservation_no = reservation.reservation_no) LEFT JOIN ticket ON schedule.ticket_no = ticket.ticket_no) ORDER BY schedule.schedule_id DESC;
```

schedule_id	time_in	time_out	reservation_no	dep_time	arr_time	ticket_no	dep_time	arr_time
6	24:00:00	21:00:00	7	21:00:00	24:00:00	1004	14:50:00	16:20:00
5	22:30:00	21:00:00	10	21:00:00	22:30:00	1002	14:50:00	16:20:00
4	03:00:00	21:00:00	8	21:00:00	23:00:00	1004	14:50:00	16:20:00
3	17:20:00	16:50:00	1	12:00:00	13:30:00	1005	16:50:00	17:20:00
2	13:30:00	12:00:00	2	12:00:00	13:30:00	1005	16:50:00	17:20:00
1	15:30:00	14:50:00	5	14:00:00	15:30:00	1000	14:50:00	15:30:00

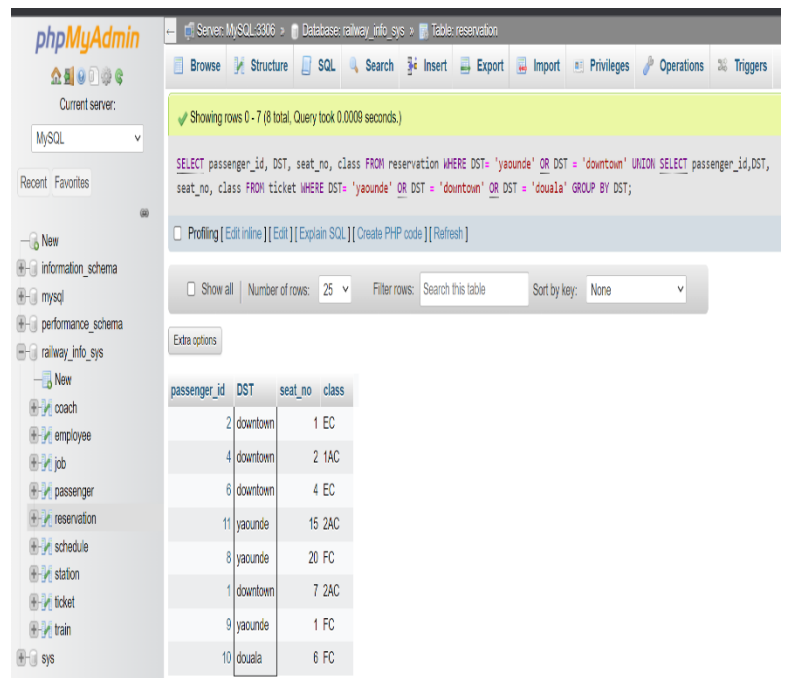
2. The query below shows a union of two different tables. In this query, I used Union operator and Where and Group By clauses;



Showing rows 0 - 15 (16 total, Query took 0.0009 seconds.)

```
SELECT DST, seat_no, class FROM reservation UNION SELECT DST, seat_no, class FROM ticket GROUP BY DST;
```

DST	seat_no	class
downtown	1	EC
downtown	2	1AC
oho	3	FC
downtown	4	EC
oho	1	FC
yaounde	15	2AC
yaounde	20	FC
douala	10	FC
limbe	14	1AC
bare	9	EC
mbanga	7	FC
oho	8	EC
calfort	9	EC
downtown	7	2AC
yaounde	1	FC
douala	6	FC

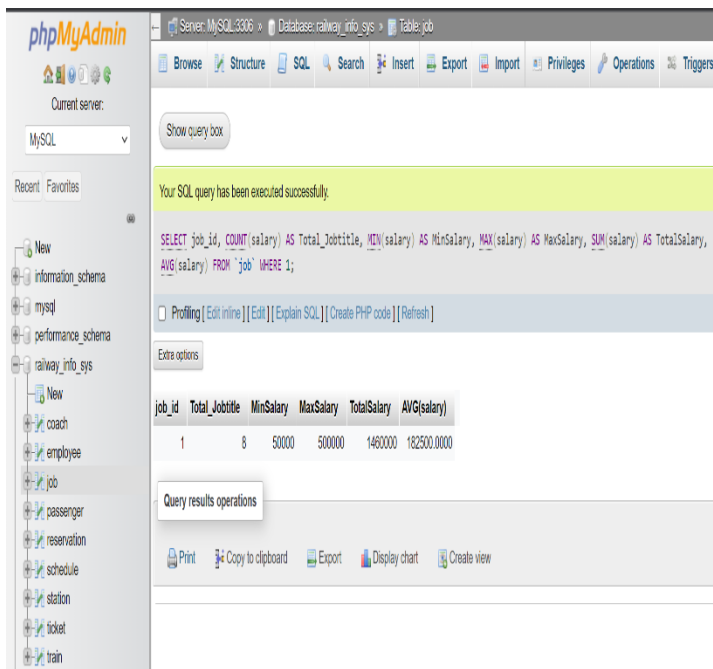


Showing rows 0 - 7 (8 total, Query took 0.0008 seconds.)

```
SELECT passenger_id, DST, seat_no, class FROM reservation WHERE DST= 'yaounde' OR DST = 'downtown' UNION SELECT passenger_id, DST, seat_no, class FROM ticket WHERE DST= 'yaounde' OR DST = 'downtown' OR DST = 'douala' GROUP BY DST;
```

passenger_id	DST	seat_no	class
2	downtown	1	EC
4	downtown	2	1AC
6	downtown	4	EC
11	yaounde	15	2AC
8	yaounde	20	FC
1	downtown	7	2AC
9	yaounde	1	FC
10	douala	6	FC

3. This query brings out the total number of job title, minimum, maximum and sum as well as average of the employee's salary. In this query, I also brought out all the employees name that first name started with "a" and the second letter was "a" using the wildcards character.

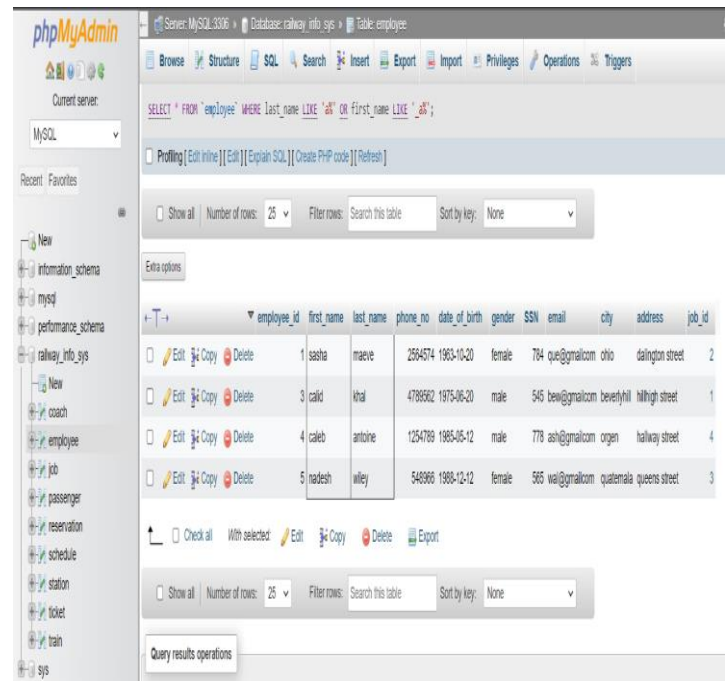


Server: MySQL3306 » Database: railway_info_sys » Table: job

Your SQL query has been executed successfully.

```
SELECT job_id, COUNT(*) AS Total_Jobtitle, MIN(salary) AS MinSalary, MAX(salary) AS MaxSalary, SUM(salary) AS TotalSalary, AVG(salary) FROM `job` WHERE 1;
```

job_id	Total_Jobtitle	MinSalary	MaxSalary	TotalSalary	AVG(salary)
1	8	50000	500000	1460000	182500.0000

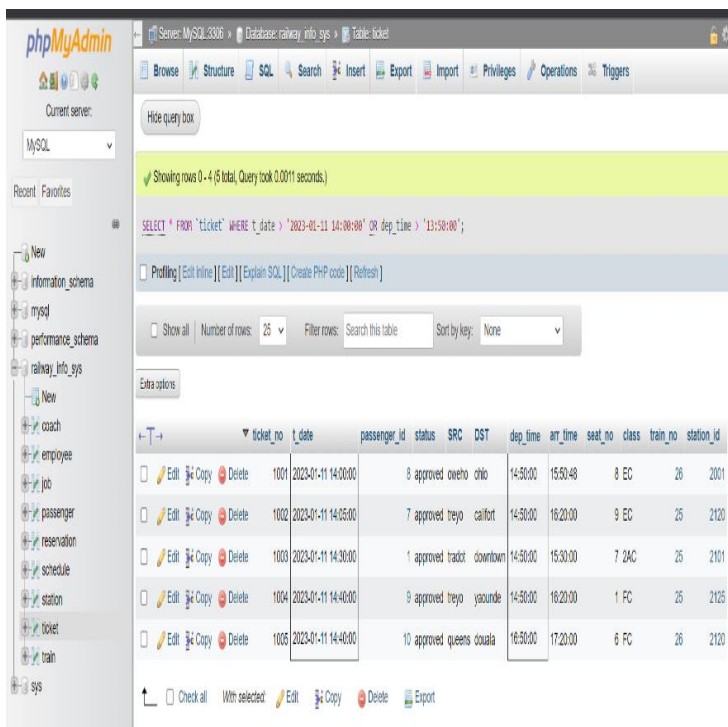


Server: MySQL3306 » Database: railway_info_sys » Table: employee

```
SELECT * FROM `employee` WHERE last_name LIKE 'a%' OR first_name LIKE 'a%';
```

employee_id	first_name	last_name	phone_no	date_of_birth	gender	SSN	email	city	address	job_id
1	sasha	maeve	2564574	1963-10-20	female	704	que@gmail.com	ohio	dalington street	2
3	calid	khali	4789502	1975-06-20	male	545	ben@gmail.com	beverlyhill	hillhigh street	1
4	caleb	antoine	1254789	1985-05-12	male	778	ash@gmail.com	orgen	halfway street	4
5	nadesh	wiley	548866	1988-12-12	female	565	wai@gmail.com	quaterlaia	queens street	3

4. This query present time which is greater than a certain period. It also brings out the **rush hours** of the company. I used the WHERE and OR clauses.

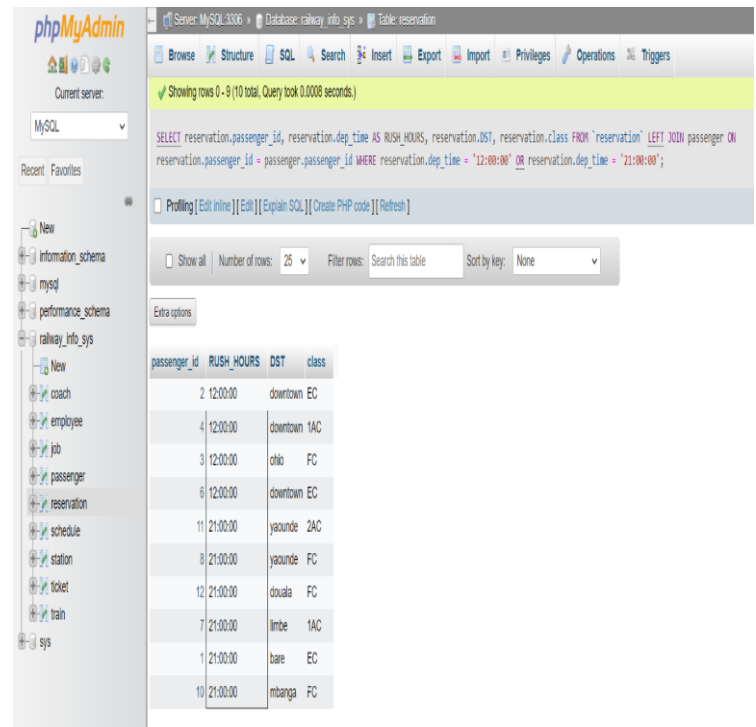


Server: MySQL3306 » Database: railway_info_sys » Table: ticket

Showing rows 0 - 4 (5 total, Query took 0.0011 seconds.)

```
SELECT * FROM `ticket` WHERE t_date > '2023-01-11 14:00:00' OR dep_time > '13:58:00';
```

ticket_no	t_date	passenger_id	status	SRC	DST	dep_time	arr_time	seat_no	class	train_no	station_id
1001	2023-01-11 14:00:00	8	approved	oweho	ohio	14:50:00	15:50:48	8	EC	26	2001
1002	2023-01-11 14:05:00	7	approved	trejo	califort	14:50:00	16:20:00	9	EC	25	2120
1003	2023-01-11 14:30:00	1	approved	tradit	downtown	14:50:00	15:30:00	7	2AC	25	2101
1004	2023-01-11 14:40:00	9	approved	trejo	yaounde	14:50:00	16:20:00	1	FC	25	2125
1005	2023-01-11 14:40:00	10	approved	queens	douala	16:50:00	17:20:00	6	FC	26	2120



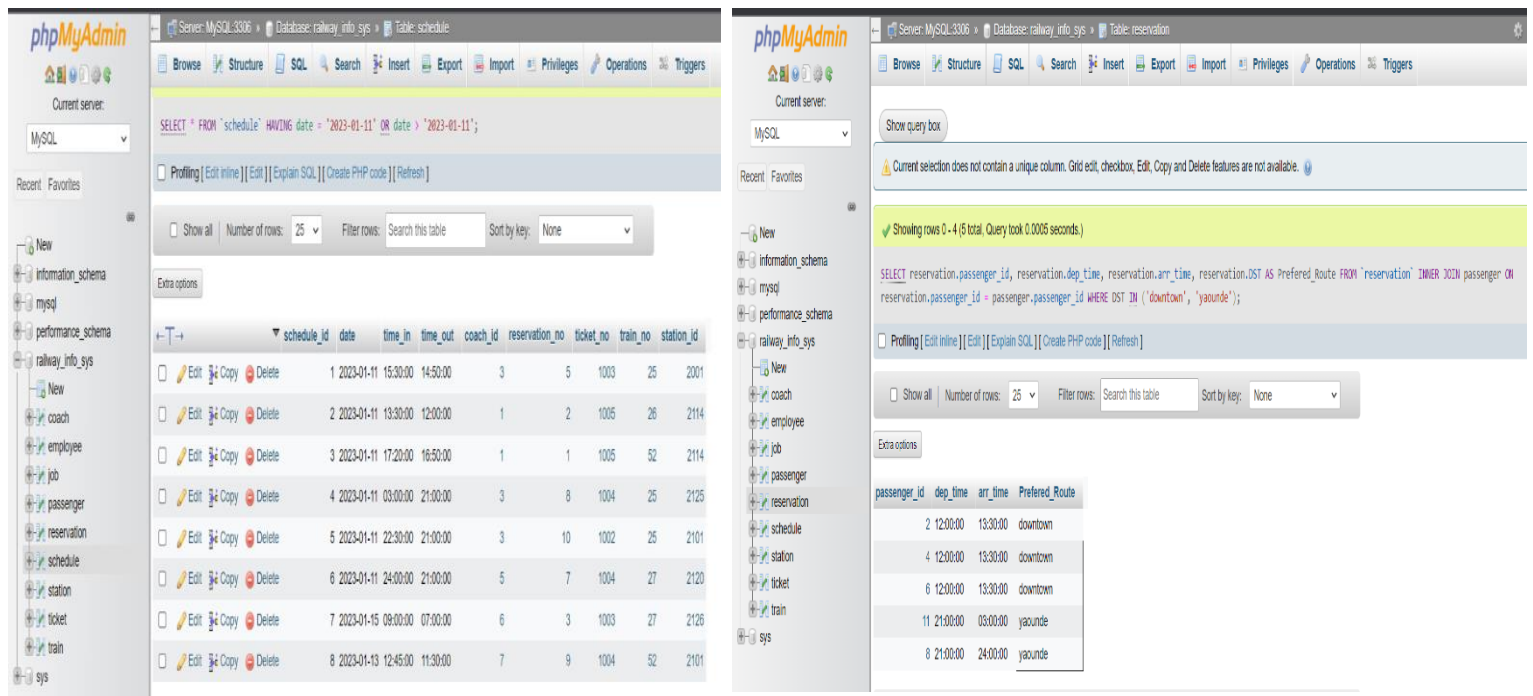
Server: MySQL3306 » Database: railway_info_sys » Table: reservation

Showing rows 0 - 9 (10 total, Query took 0.0008 seconds.)

```
SELECT reservation.passenger_id, reservation.dep_time AS RUSH_HOURS, reservation.DST, reservation.class FROM `reservation` LEFT JOIN passenger ON reservation.passenger_id = passenger.passenger_id WHERE reservation.dep_time > '12:00:00' OR reservation.dep_time > '21:00:00';
```

passenger_id	RUSH_HOURS	DST	class
2	12:00:00	downtown	EC
4	12:00:00	downtown	1AC
3	12:00:00	ohio	FC
6	12:00:00	downtown	EC
11	21:00:00	yaounde	2AC
8	21:00:00	yaounde	FC
12	21:00:00	douala	FC
7	21:00:00	limbe	1AC
1	21:00:00	bare	EC
10	21:00:00	mbanga	FC

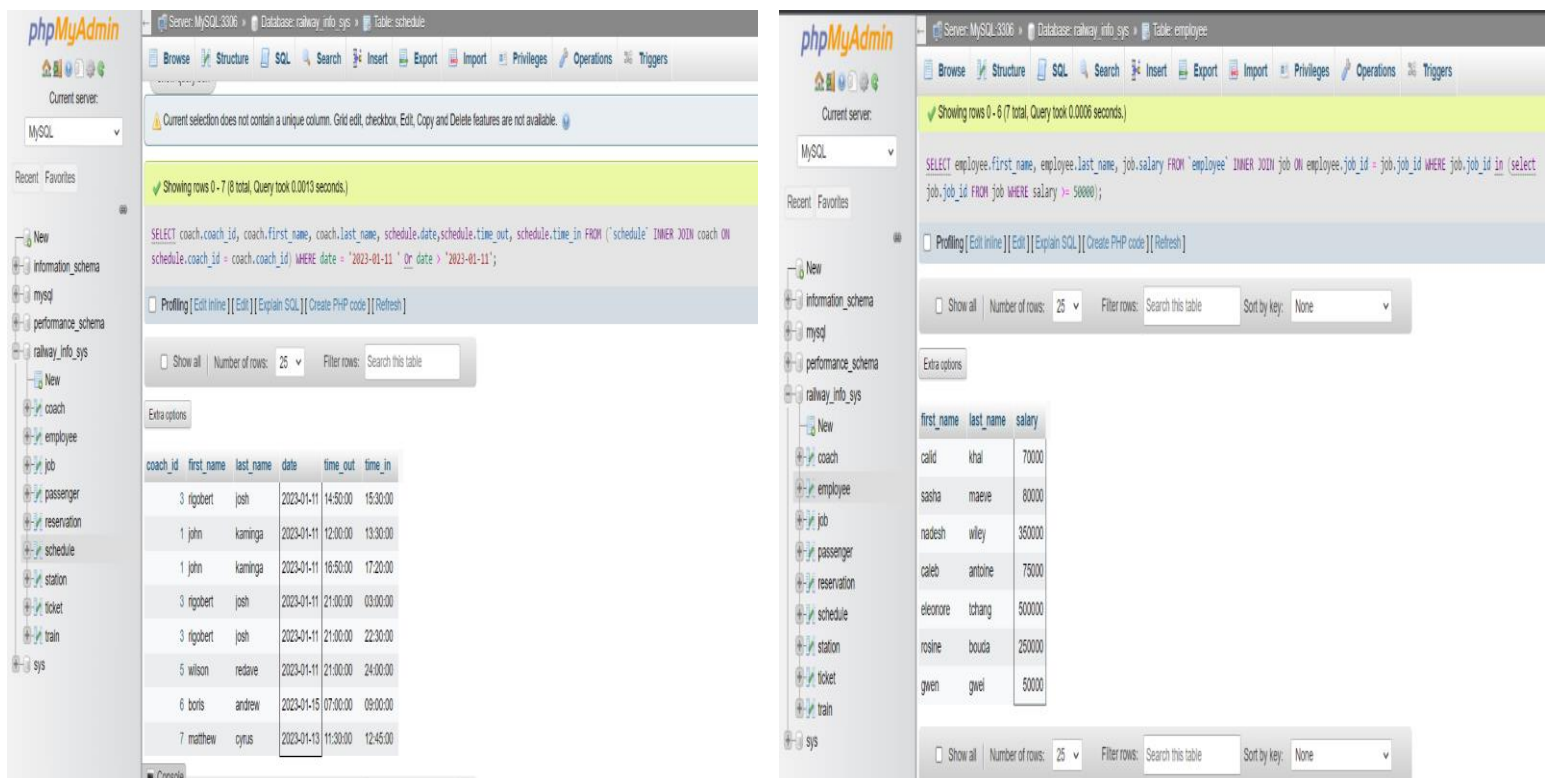
5. This query brings out the **busy schedule** of the company between other days. It also shows the preferred routes used by trains.



The first screenshot shows the 'schedule' table in the 'railway_info_sys' database. The query is: `SELECT * FROM 'schedule' HAVING date = '2023-01-11' OR date = '2023-01-11';`. The results show 8 rows of train schedules with columns: schedule_id, date, time_in, time_out, coach_id, reservation_no, ticket_no, train_no, and station_id.

The second screenshot shows the 'reservation' table. The query is: `SELECT reservation.passenger_id, reservation.dep_time, reservation.arr_time, reservation.DST AS Preferred_Route FROM 'reservation' INNER JOIN passenger ON reservation.passenger_id = passenger.passenger_id WHERE DST IN ('downtown', 'yaounde');`. The results show 8 rows of reservations with columns: passenger_id, dep_time, arr_time, and Preferred_Route.

6. This query brings out the **coaches working hours, availability on a certain day and time**. Here, I used **subquery** and a **join** query to retrieve employee's salary.



The first screenshot shows the 'schedule' table. The query is: `SELECT coach.coach_id, coach.first_name, coach.last_name, schedule.date, schedule.time_out, schedule.time_in FROM 'schedule' INNER JOIN coach ON schedule.coach_id = coach.coach_id WHERE date = '2023-01-11' OR date = '2023-01-11';`. The results show 7 rows of coach schedules with columns: coach_id, first_name, last_name, date, time_out, and time_in.

The second screenshot shows the 'employee' table. The query is: `SELECT employee.first_name, employee.last_name, job.salary FROM 'employee' INNER JOIN job ON employee.job_id = job.job_id WHERE job.job_id IN (select job.job_id FROM job WHERE salary >= 50000);`. The results show 7 rows of employee salaries with columns: first_name, last_name, and salary.

7. This query brings out the salary of employees between certain ranges by using the **BETWEEN** and **AND** clauses. Here, I also used the Exist operator to find the existence of records.

The screenshot shows the phpMyAdmin interface for the 'railway_info_sys' database. The 'employee' table is selected. The SQL query is: `SELECT employee.first_name, employee.last_name, job.salary FROM 'employee' INNER JOIN job ON employee.job_id = job.job_id WHERE job.job_id IN (select job.job_id FROM job WHERE salary BETWEEN 150000 AND 600000);` The results show three rows of employee data.

first_name	last_name	salary
nadesh	wiley	350000
eleonore	tchang	500000
rosine	bouda	250000

The screenshot shows the phpMyAdmin interface for the 'railway_info_sys' database. The 'employee' table is selected. The SQL query is: `SELECT employee.first_name, employee.last_name, job.salary, job.hired_date, employee.gender, employee.city FROM 'employee' INNER JOIN job ON employee.job_id = job.job_id WHERE EXISTS (select job.job_id FROM job WHERE hired_date > '2001-06-25') GROUP BY employee.city;` The results show three rows of employee data grouped by city.

first_name	last_name	salary	hired_date	gender	city
calid	khal	70000	2012-10-12	male	beverlyhill
sasha	maeve	80000	2018-01-20	female	ohio
nadesh	wiley	350000	2010-05-02	female	ogen

8. This query retrieve the schedule of a given date and time which presents the passenger_id, coach name, DST, train_no and station_name by using subqueries and join queries.

The screenshot shows the phpMyAdmin interface for the 'railway_info_sys' database. The 'schedule' table is selected. The SQL query is: `SELECT schedule.schedule_id, schedule.date, schedule.time_out, schedule.time_in, coach.last_name AS Coach_Name, reservation.passenger_id, reservation.DST AS reservation_DST, ticket.DST AS ticket_DST, train.train_no, station.station_name FROM 'schedule' INNER JOIN coach ON schedule.coach_id = coach.coach_id INNER JOIN reservation ON schedule.reservation_no = reservation.reservation_no INNER JOIN ticket ON schedule.ticket_no = ticket.ticket_no INNER JOIN station ON schedule.station_id = station.station_id INNER JOIN train ON schedule.train_no = train.train_no where schedule.reservation_no in (SELECT reservation_no FROM reservation inner JOIN passenger ON reservation.passenger_id = passenger.passenger_id WHERE reservation.DST in ('downtown', 'yaounde', 'douala', 'ohio')) OR schedule.ticket_no in (SELECT ticket_no FROM ticket [...])` The results show six rows of schedule data.

schedule_id	date	time_out	time_in	Coach_Name	passenger_id	reservation_DST	ticket_DST	train_no	station_name
1	2023-01-11	14:50:00	15:30:00	josh	5	ohio	downtown	25	Westbrone
4	2023-01-11	21:00:00	03:00:00	josh	12	douala	yaounde	25	downside
5	2023-01-11	21:00:00	22:30:00	josh	1	ohio	downtown	25	Bronx
2	2023-01-11	12:00:00	13:30:00	kaminga	4	downtown	douala	26	Trinity
6	2023-01-11	21:00:00	24:00:00	redave	8	yaounde	yaounde	27	Dracant
3	2023-01-11	16:50:00	17:20:00	kaminga	2	downtown	douala	52	Trinity