At Your Door

A Home Service Booking System

ISM6218.001F22.88140 Advanced Database Management

UNIVERSITY of SOUTH FLORIDA

Group 5

Anirudh Voruganti

Ankit Patro

Bharathi Pemmasani

Sri Guru Achuth Gadicherla

Topic	***Description***	***Points***	***Team Member***
Database Design	This phase focuses on logical database architecture using ERD and normalization to manage redundancy and integrity requirements for data quality and define the ERD to accommodate future project upgrades.	25	Anirudh, Ankit, Bharathi, Achuth.
Query Writing	Create SQL queries to deliver user suggestions based on the specified use cases and database programming for stored procedures.	25	Anirudh, Ankit, Bharathi, Achuth.
Performance Tuning	Tuning performance and query optimization for future scope.	25	Anirudh, Ankit, Bharathi, Achuth.
Other Topics	Implement Database security, Data visualization, and integration with apps.	25	Anirudh, Ankit, Bharathi, Achuth.

Table of Contents

Executive Summary

We often have hectic schedules since we have so much to achieve and learn. Some of us are swamped with work, while others are kept busy with domestic duties, family obligations, everyday jobs, and other things. In the middle of this situation, we neglect to provide self-care, house care, and other diverse requirements even a minute of our time.

With the evolution of this world into a technologically advanced society, we can complete tasks with a single tap. Home services are the most popular choice among app users. These are the contemporary answers to age-old problems.

Today, we can catch up with varied on-demand home service applications in the mobile app market that can assist us in fulfilling all such types of requirements anytime, anywhere.

With this project, we intend to build a backend relational database for an online service booking platform that matches customers' requirements with professionals who can fulfill their service needs. The platform aims to enable users to find and hire the right professionals, such as carpenters, plumbers, electricians, etc., for all their needs.

The primary entities considered to design the database to facilitate the above-mentioned use case are users, locations, reviews, service variants, categories, prices, availability, or business hours and service descriptions.

List of Tables Created

Table	Description	Rows	Type	Collation	Size (KiB)
customers	Table containing details of the customers	2,000	InnoDB	latin1_swedish_ci	256.0
customer_bookings	Table containing booking details	2,456	InnoDB	latin1_swedish_ci	320.0
customer_booking_partner_allocation	Table containing details about partner to customer allocation	2,456	InnoDB	utf8mb4_general_ci	352.0
customer_reviews	Table containing reviews for service partner	1,653	InnoDB	utf8mb4_general_ci	144.0
partners	Tables containing service partner details	300	InnoDB	utf8mb4_general_ci	80.0
partner_service_categories	Table containing partner service categories	425	InnoDB	utf8mb4_general_ci	80.0
partner_service_mapping	Table containing partner service mappings	1,387	InnoDB	utf8mb4_general_ci	240.0
service_categories	Table containing different service categories	21	InnoDB	utf8mb4_general_ci	16.0
service_names	Table containing different service names	70	InnoDB	utf8mb4_general_ci	32.0
9 tables	**Sum**	**10,768**	**InnoDB**	**utf8mb4_general_ci**	**1.5 MiB**

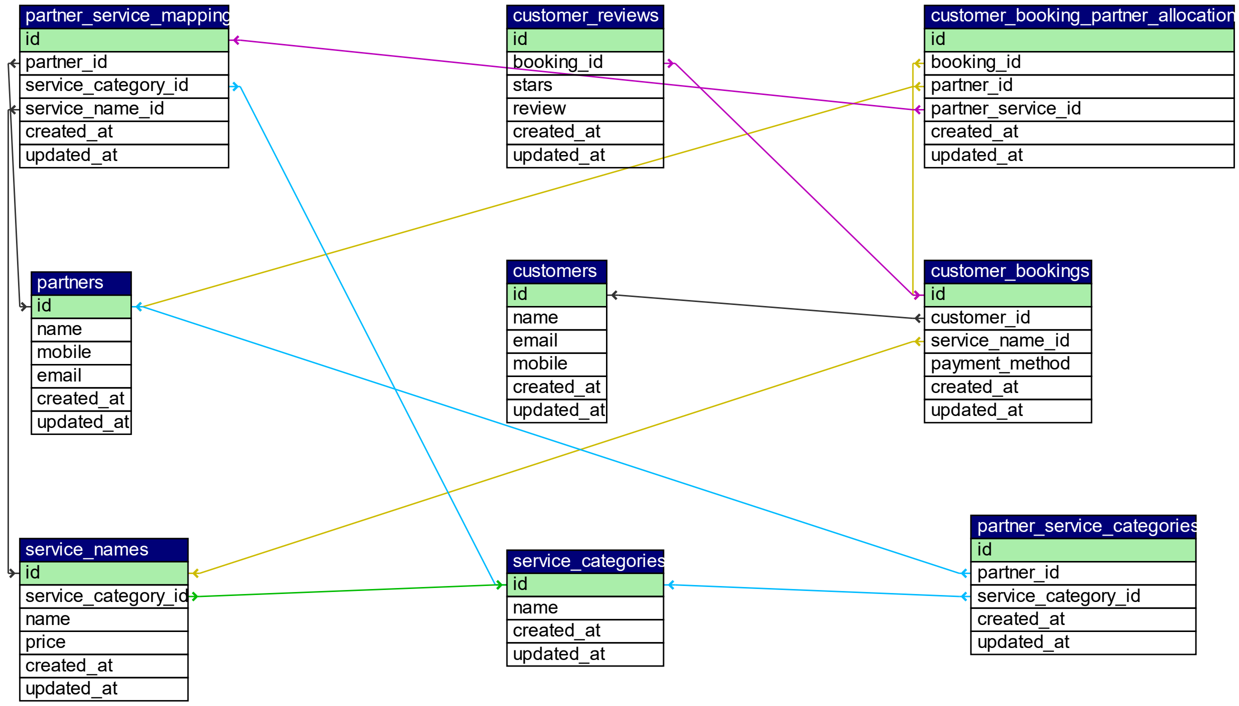
Database Design

To build this database we have utilised XAMPP server locally, The Apache HTTP Server, MariaDB database are the core components of XAMPP, an open-source cross-platform web server stack bundle created by Apache Friends. It is possible to go from a local test server to a live server because most real-world web server deployments employ the same components as XAMPP.

MariaDB 10.4.25 version has been utilized for hosting this database, The biggest open-source database community is MySQL. Since MariaDB is a derivative of MySQL, it is entirely backward compatible.

The MySQL relational database management system (RDBMS) has been forked into MariaDB by the community to keep it free and open-source software under the GNU General Public License. While MySQL's future is uncertain, MariaDB's mission statement is still open source and cross-platform.

Logical Entity Relationship (ER) Diagram



Data Dictionary

Table Name:	**customers**			
Column	**Type**	**Null**	**Default**	**Links to**
id (Primary)	int(10)	No		
name	varchar(45)	No		
email	varchar(45)	No		
mobile	varchar(15)	No		
created_at	timestamp	No	current_timestamp()	
updated_at	timestamp	No	current_timestamp()	

Table Name:	**customer_bookings**			
Column	**Type**	**Null**	**Default**	**Links to**
id (Primary)	int(10)	No		
customer_id	int(10)	No		customers -> id
service_name_id	int(10)	No		service_names -> id
payment_method	enum('CREDIT_CARD', 'DEBIT_CARD', 'CASH')	No		
created_at	timestamp	No	current_timestamp()	
updated_at	timestamp	No	current_timestamp()	

Table Name:	**customer_booking_partner_allocation**			
Column	**Type**	**Null**	**Default**	**Links to**
id (Primary)	int(11)	No		
booking_id	int(11)	No		customer_bookings -> id
partner_id	int(11)	No		partners -> id
partner_service_id	int(11)	No		partner_service_mapping -> id
created_at	timestamp	No	current_timestamp()	
updated_at	timestamp	No	current_timestamp()	

Table Name:	**customer_reviews**			
Column	**Type**	**Null**	**Default**	**Links to**
id (Primary)	int(11)	No		
booking_id	int(11)	No		customer_bookings -> id
stars	int(11)	No		
review	text	Yes	NULL	
created_at	timestamp	No	current_timestamp()	
updated_at	timestamp	No	current_timestamp()	

Table Name:	**partners**

Column	**Type**	**Null**	**Default**	**Links to**
id (Primary)	int(11)	No		
name	tinytext	No		
mobile	tinytext	No		
email	varchar(50)	Yes	NULL	
created_at	timestamp	No	current_timestamp()	
updated_at	timestamp	No	current_timestamp()	

Table Name:	**partner_service_categories**			
Column	**Type**	**Null**	**Default**	**Links to**
id (Primary)	int(11)	No		
partner_id	int(11)	No		partners -> id
service_category_id	int(11)	No		service_categories -> id
created_at	timestamp	No	current_timestamp()	
updated_at	timestamp	No	current_timestamp()	

Table Name:	**partner_service_mapping**			
Column	**Type**	**Null**	**Default**	**Links to**
id (Primary)	int(11)	No		
partner_id	int(11)	No		partners -> id
service_category_id	int(11)	No		service_categories -> id
service_name_id	int(11)	No		service_names -> id
created_at	timestamp	No	current_timestamp()	
updated_at	timestamp	No	current_timestamp()	

Table Name:	**service_categories**			
Column	**Type**	**Null**	**Default**	**Links to**
id (Primary)	int(11)	No		
name	text	No		
created_at	timestamp	No	current_timestamp()	
updated_at	timestamp	No	current_timestamp()	

Table Name:	**service_names**			
Column	**Type**	**Null**	**Default**	**Links to**
id (Primary)	int(11)	No		
service_category_id	int(11)	No		service_categories -> id
name	text	No		
price	int(11)	No		
created_at	timestamp	No	current_timestamp()	
updated_at	timestamp	No	current_timestamp()	

Queries Used for Creating Tables with constraints for Data Integrity

```
--
-- Table structure for table `customers`
--
DROP TABLE IF EXISTS `customers`;
CREATE TABLE `customers` (
  `id` int(10) NOT NULL,
  `name` varchar(45) NOT NULL,
  `email` varchar(45) NOT NULL,
  `mobile` varchar(15) NOT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  `updated_at` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- AUTO_INCREMENT for table `customers`
--
ALTER TABLE `customers`
  MODIFY `id` int(10) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2031;


--
-- Table structure for table `customer_bookings`
--

DROP TABLE IF EXISTS `customer_bookings`;
CREATE TABLE `customer_bookings` (
  `id` int(10) NOT NULL,
  `customer_id` int(10) NOT NULL,
  `service_name_id` int(10) NOT NULL,
  `payment_method` enum('CREDIT_CARD','DEBIT_CARD','CASH') NOT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  `updated_at` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Constraints for table `customer_bookings`
--
ALTER TABLE `customer_bookings`
  ADD CONSTRAINT `fk_books_cust_id` FOREIGN KEY (`customer_id`) REFERENCES
`customers` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,
  ADD CONSTRAINT `fk_books_serv_name_id` FOREIGN KEY (`service_name_id`)
REFERENCES `service_names` (`id`) ON DELETE CASCADE ON UPDATE CASCADE;

--
-- AUTO_INCREMENT for table `customer_bookings`
--
ALTER TABLE `customer_bookings`
  MODIFY `id` int(10) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2457;

--
-- Table structure for table `customer_booking_partner_allocation`
--
```

```
DROP TABLE IF EXISTS `customer_booking_partner_allocation`;
CREATE TABLE `customer_booking_partner_allocation` (
  `id` int(11) NOT NULL,
  `booking_id` int(11) NOT NULL,
  `partner_id` int(11) NOT NULL,
  `partner_service_id` int(11) NOT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  `updated_at` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;


--
-- Constraints for table `customer_booking_partner_allocation`
--
ALTER TABLE `customer_booking_partner_allocation`
  ADD CONSTRAINT `customer_booking_partner_allocation_ibfk_1` FOREIGN KEY
(`booking_id`) REFERENCES `customer_bookings` (`id`),
  ADD CONSTRAINT `customer_booking_partner_allocation_ibfk_2` FOREIGN KEY
(`partner_id`) REFERENCES `partners` (`id`),
  ADD CONSTRAINT `customer_booking_partner_allocation_ibfk_3` FOREIGN KEY
(`partner_service_id`) REFERENCES `partner_service_mapping` (`id`);

--
-- AUTO_INCREMENT for table `customer_booking_partner_allocation`
--
ALTER TABLE `customer_booking_partner_allocation`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2457;

--
-- Table structure for table `customer_reviews`
--

DROP TABLE IF EXISTS `customer_reviews`;
CREATE TABLE `customer_reviews` (
  `id` int(11) NOT NULL,
  `booking_id` int(11) NOT NULL,
  `stars` int(11) NOT NULL,
  `review` text DEFAULT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  `updated_at` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;



--
-- Constraints for table `customer_reviews`
--
ALTER TABLE `customer_reviews`
  ADD CONSTRAINT `customer_reviews_ibfk_5` FOREIGN KEY (`booking_id`)
REFERENCES `customer_bookings` (`id`);
--

--
-- AUTO_INCREMENT for table `customer_reviews`
--
```

```
ALTER TABLE `customer_reviews`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2457;

-- Table structure for table `partners`
--

DROP TABLE IF EXISTS `partners`;
CREATE TABLE `partners` (
  `id` int(11) NOT NULL,
  `name` tinytext NOT NULL,
  `mobile` tinytext NOT NULL,
  `email` varchar(50) DEFAULT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  `updated_at` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- AUTO_INCREMENT for table `partners`
--
ALTER TABLE `partners`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=321;

--
-- Table structure for table `partner_service_categories`
--

DROP TABLE IF EXISTS `partner_service_categories`;
CREATE TABLE `partner_service_categories` (
  `id` int(11) NOT NULL,
  `partner_id` int(11) NOT NULL,
  `service_category_id` int(11) NOT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  `updated_at` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Constraints for table `partner_service_categories`
--
ALTER TABLE `partner_service_categories`
  ADD CONSTRAINT `partner_service_categories_ibfk_1` FOREIGN KEY
(`partner_id`) REFERENCES `partners` (`id`),
  ADD CONSTRAINT `partner_service_categories_ibfk_2` FOREIGN KEY
(`service_category_id`) REFERENCES `service_categories` (`id`);

--
-- AUTO_INCREMENT for table `partner_service_categories`
--
ALTER TABLE `partner_service_categories`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=1776;

--
-- Table structure for table `partner_service_mapping`
--

DROP TABLE IF EXISTS `partner_service_mapping`;
```

```
CREATE TABLE `partner_service_mapping` (
  `id` int(11) NOT NULL,
  `partner_id` int(11) NOT NULL,
  `service_category_id` int(11) NOT NULL,
  `service_name_id` int(11) NOT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  `updated_at` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Constraints for table `partner_service_mapping`
--
ALTER TABLE `partner_service_mapping`
  ADD CONSTRAINT `partner_service_mapping_ibfk_1` FOREIGN KEY
(`partner_id`) REFERENCES `partners` (`id`),
  ADD CONSTRAINT `partner_service_mapping_ibfk_3` FOREIGN KEY
(`service_name_id`) REFERENCES `service_names` (`id`),
  ADD CONSTRAINT `partner_service_mapping_ibfk_4` FOREIGN KEY
(`service_category_id`) REFERENCES `service_categories` (`id`);

--
-- AUTO_INCREMENT for table `partner_service_mapping`
--
ALTER TABLE `partner_service_mapping`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=1554;

--
-- Table structure for table `service_categories`
--

DROP TABLE IF EXISTS `service_categories`;
CREATE TABLE `service_categories` (
  `id` int(11) NOT NULL,
  `name` text NOT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  `updated_at` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- AUTO_INCREMENT for table `service_categories`
--
ALTER TABLE `service_categories`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=25;

--
-- Table structure for table `service_names`
--

DROP TABLE IF EXISTS `service_names`;
CREATE TABLE `service_names` (
  `id` int(11) NOT NULL,
  `service_category_id` int(11) NOT NULL,
  `name` text NOT NULL,
  `price` int(11) NOT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  `updated_at` timestamp NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Constraints for table `service_names`
--
ALTER TABLE `service_names`
  ADD CONSTRAINT `service_names_ibfk_1` FOREIGN KEY (`service_category_id`)
REFERENCES `service_categories` (`id`);
COMMIT;


--
-- AUTO_INCREMENT for table `service_names`
--
ALTER TABLE `service_names`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=95;
```

Query Writing

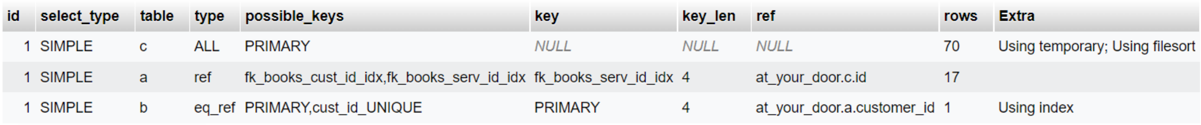
Query 1: Top 10 most booked services

```
127.0.0.1/at_your_door/c/
      http://localhost/phpmyadmin/index.php?route=/database/sql&db=at_your_
door
   Showing rows 0 -  9 (10 total, Query took 0.0170 seconds.)
```

```
SELECT count(*),c.name FROM `customer_bookings` a join customers b on
a.customer_id=b.id join service_names c on a.service_name_id=c.id group by
c.name order by count(*) desc limit 10;
```

```
count(*)     name
81     Cupboard
78     Modular Kitchen
78     Tiling
73     Geyser Installation
68     False Ceiling
60     Furniture Work
55     Desktop Repair
52     Paint
49     Geyser Uninstallation
49     Small Geyser for Kitchen Installation
```

Explain plan of the query



id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	c	ALL	PRIMARY	NULL	NULL	NULL	70	Using temporary; Using filesort
1	SIMPLE	a	ref	fk_books_cust_id_idx,fk_books_serv_id_idx	fk_books_serv_id_idx	4	at_your_door.c.id	17	
1	SIMPLE	b	eq_ref	PRIMARY,cust_id_UNIQUE	PRIMARY	4	at_your_door.a.customer_id	1	Using index

Query 2: Top 10 partners with 5 start rating

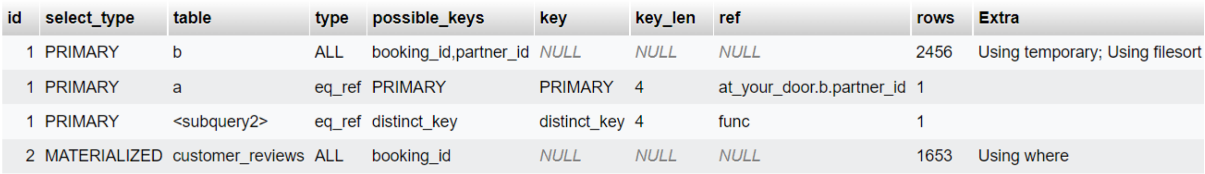
```
 127.0.0.1/at_your_door/partners/
      http://localhost/phpmyadmin/index.php?route=/database/sql&db=at_your_
door
   Showing rows 0 -  9 (10 total, Query took 0.0138 seconds.)
```

```
select name,count(*) from partners a join
customer_booking_partner_allocation b on a.id=b.partner_id where
b.booking_id in (select booking_id from customer_reviews where stars=5)
group by name limit 10;
```

```
name  count(*)
Angela Langworth  24
Clint Becker      17
Cristina Gerhold  12
Derek Marvin      4
Dr. Darryl Deckow 5
```

```
Ezequiel Upton     35
Fredrick Funk      2
Jaylen Armstrong V       15
Jesus Hagenes      31
Jose Muller 6
```

Explain plan of the query



id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	PRIMARY	b	ALL	booking_id,partner_id	*NULL*	*NULL*	*NULL*	2456	Using temporary; Using filesort
1	PRIMARY	a	eq_ref	PRIMARY	PRIMARY	4	at_your_door.b.partner_id	1	
1	PRIMARY	<subquery2>	eq_ref	distinct_key	distinct_key	4	func	1	
2	MATERIALIZED	customer_reviews	ALL	booking_id	*NULL*	*NULL*	*NULL*	1653	Using where

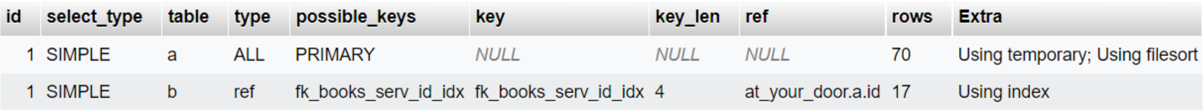
Query 3: Top 5 services with highest revenue yield

```
127.0.0.1/at_your_door/service_names/
      http://localhost/phpmyadmin/index.php?route=/database/sql&db=at_your_
door
   Showing rows 0 -  4 (5 total, Query took 0.0150 seconds.)
```

```
select sum(price),name from service_names a join customer_bookings b on
a.id=b.service_name_id group by name order by sum(price) desc limit 5;
```

```
sum(price)        name
1564  False Ceiling
1500  Furniture Work
1197  Tiling
1183  Geyser Uninstallation
1072  Paint
```

Explain plan of the query



id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	a	ALL	PRIMARY	*NULL*	*NULL*	*NULL*	70	Using temporary; Using filesort
1	SIMPLE	b	ref	fk_books_serv_id_idx	fk_books_serv_id_idx	4	at_your_door.a.id	17	Using index

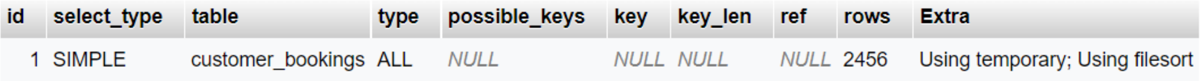
Query 4: Number of Bookings based on payment method

```
127.0.0.1/at_your_door/customer_bookings/
      http://localhost/phpmyadmin/index.php?route=/database/sql&db=at_your_
door
   Showing rows 0 -  2 (3 total, Query took 0.0018 seconds.)
```

```
select count(*),payment_method from customer_bookings group by
payment_method order by count(*) desc;
```

```
count(*)     payment_method
835    CASH
830    DEBIT_CARD
791    CREDIT_CARD
```

Explain plan of the query



id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	customer_bookings	ALL	*NULL*	*NULL*	*NULL*	*NULL*	2456	Using temporary; Using filesort

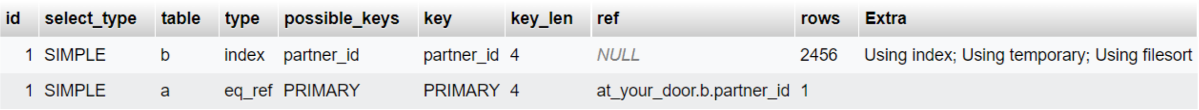
Query 5: Top 5 partners with highest number of bookings

```
127.0.0.1/at_your_door/partners/
      http://localhost/phpmyadmin/index.php?route=/database/sql&db=at_your_
door
   Showing rows 0 -  4 (5 total, Query took 0.0128 seconds.)
```

select count(*),a.name from partners a join customer_booking_partner_allocation b on a.id=b.partner_id group by a.name order by count(*) desc limit 5;

```
count(*)     name
275    Ezequiel Upton
245    Ruben Rippin
241    Miss Blanche Herzog
214    Jesus Hagenes
179    Angela Langworth
```

Explain plan of the query



id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	b	index	partner_id	partner_id	4	*NULL*	2456	Using index; Using temporary; Using filesort
1	SIMPLE	a	eq_ref	PRIMARY	PRIMARY	4	at_your_door.b.partner_id	1	

Performance Tuning

To improve the Database performance, we have implemented B-Tree Indexes in our database.

Indexes: MySql indexes function like a book's index. While book indexes provide information about the pages where a word appears, MySql indexes provide information about the rows that contain the matching data. Values from one or more table columns are contained in an index. The order of the columns is crucial if we index more than one column since MySQL can only search effectively on the index's leftmost prefix.

B-Tree Indexes: Each leaf page in a B-Tree is equally spaced from the root, and all data are kept in ascending order.

Because the storage engine does not have to go through the whole table to retrieve the required data, a B-Tree index speeds up data access. The root node is where it starts instead. The storage engine follows pointers to child nodes that are stored in the slots of the root node. The values in the node pages, which provide the upper and lower limits of the values in the child nodes, are used to determine the appropriate pointer. A leaf page is eventually reached, or the storage engine concludes that the needed value needs to be present. Because they contain pointers to the indexed data rather than links to other pages, leaf pages are unique.

Indexes for customers table

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null
PRIMARY	BTREE	Yes	No	id	2000	A	No
cust_id_UNIQUE	BTREE	Yes	No	id	2000	A	No

Indexes for customer_bookings table

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null
PRIMARY	BTREE	Yes	No	id	2456	A	No
books_id_UNIQUE	BTREE	Yes	No	id	2456	A	No
fk_books_cust_id_idx	BTREE	No	No	customer_id	2456	A	No
fk_books_serv_id_idx	BTREE	No	No	service_name_id	144	A	No

Indexes for customer_booking_partner_allocation

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null
PRIMARY	BTREE	Yes	No	id	2456	A	No
booking_id	BTREE	No	No	booking_id	2456	A	No
partner_id	BTREE	No	No	partner_id	44	A	No
partner_service_id	BTREE	No	No	partner_service_id	144	A	No

Indexes for customer_reviews table

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null
PRIMARY	BTREE	Yes	No	id	1653	A	No
booking_id	BTREE	No	No	booking_id	1653	A	No

Indexes for partners table							
Keyname	**Type**	**Unique**	**Packed**	**Column**	**Cardinality**	**Collation**	**Null**
PRIMARY	BTREE	Yes	No	id	300	A	No
email	BTREE	Yes	No	email	300	A	Yes

Indexes for partner_service_categories table							
Keyname	**Type**	**Unique**	**Packed**	**Column**	**Cardinality**	**Collation**	**Null**
PRIMARY	BTREE	Yes	No	id	425	A	No
partner_id	BTREE	Yes	No	partner_id	425	A	No
				service_category_id	425	A	No
service_category_id	BTREE	No	No	service_category_id	42	A	No

Indexes for partner_service_mapping table							
Keyname	**Type**	**Unique**	**Packed**	**Column**	**Cardinality**	**Collation**	**Null**
PRIMARY	BTREE	Yes	No	id	1387	A	No
partner_id-service_name	BTREE	Yes	No	partner_id	693	A	No
				service_name_id	1387	A	No
service_name_id	BTREE	No	No	service_name_id	154	A	No
service_category_id	BTREE	No	No	service_category_id	42	A	No

Indexes for service_categories table							
Keyname	**Type**	**Unique**	**Packed**	**Column**	**Cardinality**	**Collation**	**Null**
PRIMARY	BTREE	Yes	No	id	21	A	No

Indexes for service_names table							
Keyname	**Type**	**Unique**	**Packed**	**Column**	**Cardinality**	**Collation**	**Null**
PRIMARY	BTREE	Yes	No	id	70	A	No
service_category_id	BTREE	No	No	service_category_id	70	A	No

Queries used for creating indexes

```
--
-- Indexes for table `customers`
--
ALTER TABLE `customers`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `cust_id_UNIQUE` (`id`);

--
-- Indexes for table `customer_bookings`
--
ALTER TABLE `customer_bookings`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `books_id_UNIQUE` (`id`),
  ADD KEY `fk_books_cust_id_idx` (`customer_id`),
  ADD KEY `fk_books_serv_id_idx` (`service_name_id`);
```

```
--
-- Indexes for table `customer_booking_partner_allocation`
--
ALTER TABLE `customer_booking_partner_allocation`
  ADD PRIMARY KEY (`id`),
  ADD KEY `booking_id` (`booking_id`),
  ADD KEY `partner_id` (`partner_id`),
  ADD KEY `partner_service_id` (`partner_service_id`);

--
-- Indexes for table `customer_reviews`
--
ALTER TABLE `customer_reviews`
  ADD PRIMARY KEY (`id`),
  ADD KEY `booking_id` (`booking_id`);

--
-- Indexes for table `partners`
--
ALTER TABLE `partners`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `email` (`email`);

--
-- Indexes for table `partner_service_categories`
--
ALTER TABLE `partner_service_categories`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `partner_id` (`partner_id`,`service_category_id`),
  ADD KEY `service_category_id` (`service_category_id`);

--
-- Indexes for table `partner_service_mapping`
--
ALTER TABLE `partner_service_mapping`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `partner_id-service_name` (`partner_id`,`service_name_id`)
USING BTREE,
  ADD KEY `service_name_id` (`service_name_id`),
  ADD KEY `service_category_id` (`service_category_id`);

--
-- Indexes for table `service_categories`
--
ALTER TABLE `service_categories`
  ADD PRIMARY KEY (`id`);

--
-- Indexes for table `service_names`
--
ALTER TABLE `service_names`
  ADD PRIMARY KEY (`id`),
  ADD KEY `service_category_id` (`service_category_id`);
```

Data Visualization

We have connected out MySQL server to tableau for live data visualization using the connector "mysql-connector-odbc"

Open Database Connectivity (ODBC):

A standard application programming interface (API) for gaining access to database management systems is called Open Database Connectivity (ODBC). With just little modifications to the data access code, an ODBC-written program may be transferred to different platforms on both the client and server sides.

Through the use of an ODBC driver as a translation layer between the application and the DBMS, ODBC achieves DBMS independence. The application connects to an ODBC driver manager to use ODBC functions, and the driver management then sends the query to the DBMS.

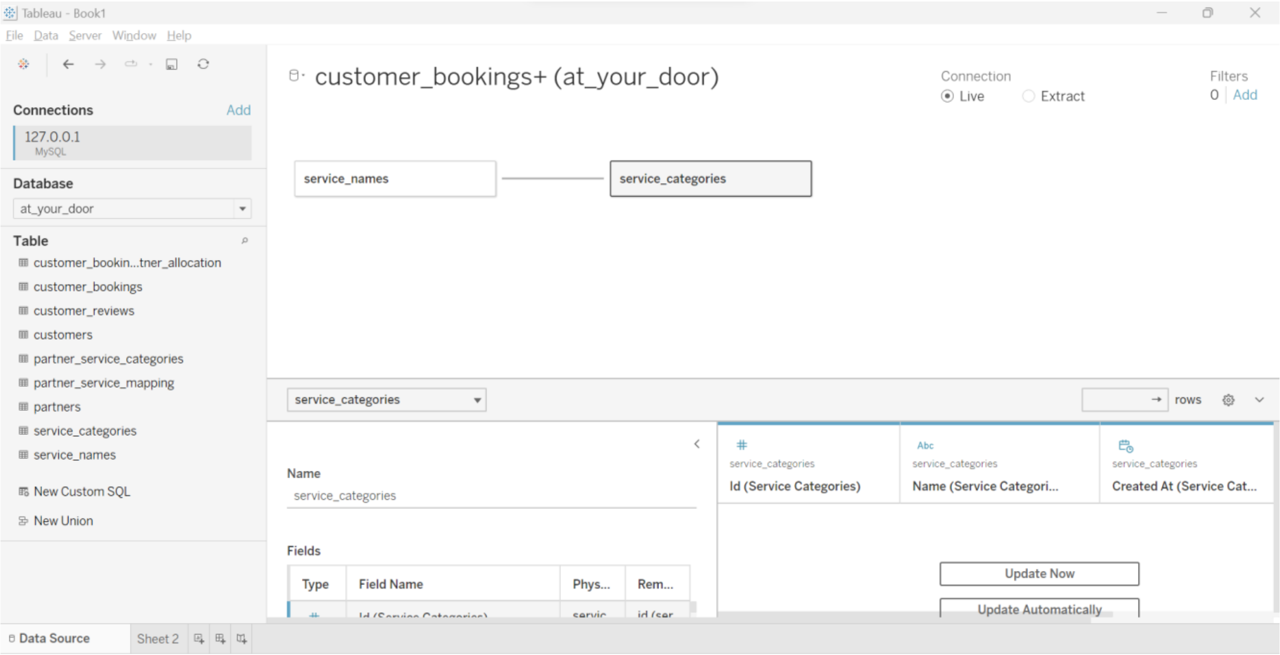


Tableau - Book1

customer_bookings+ (at_your_door)

Connection ● Live ○ Extract

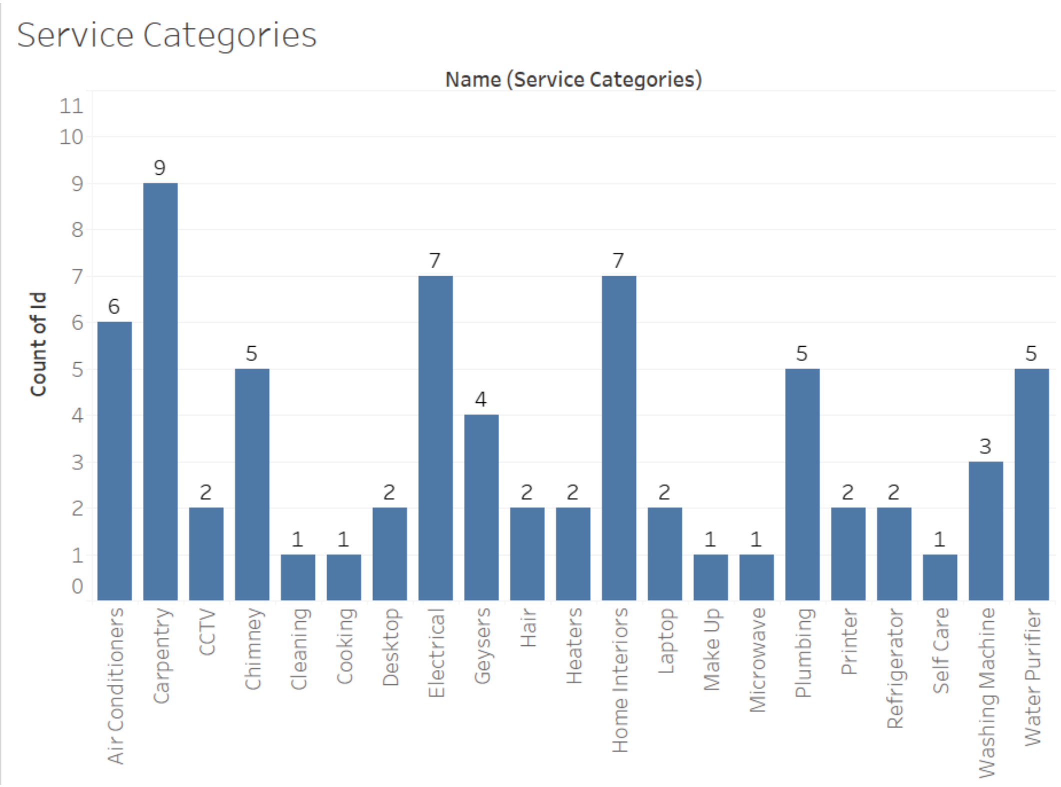
Filters 0 | Add

Connections: 127.0.0.1 MySQL

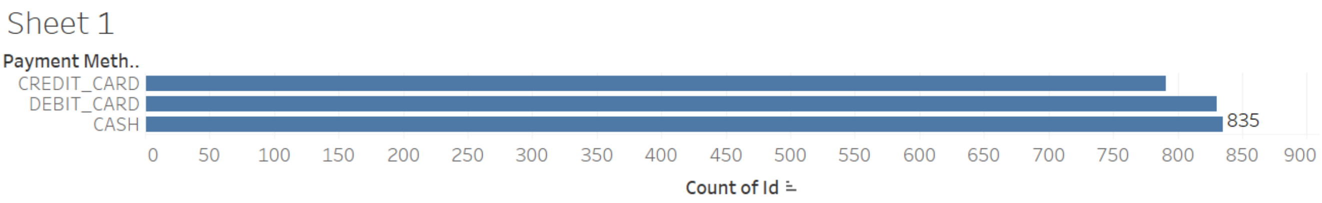
Database: at_your_door

Name: service_categories

Visualisation 1: Number of Services under each category

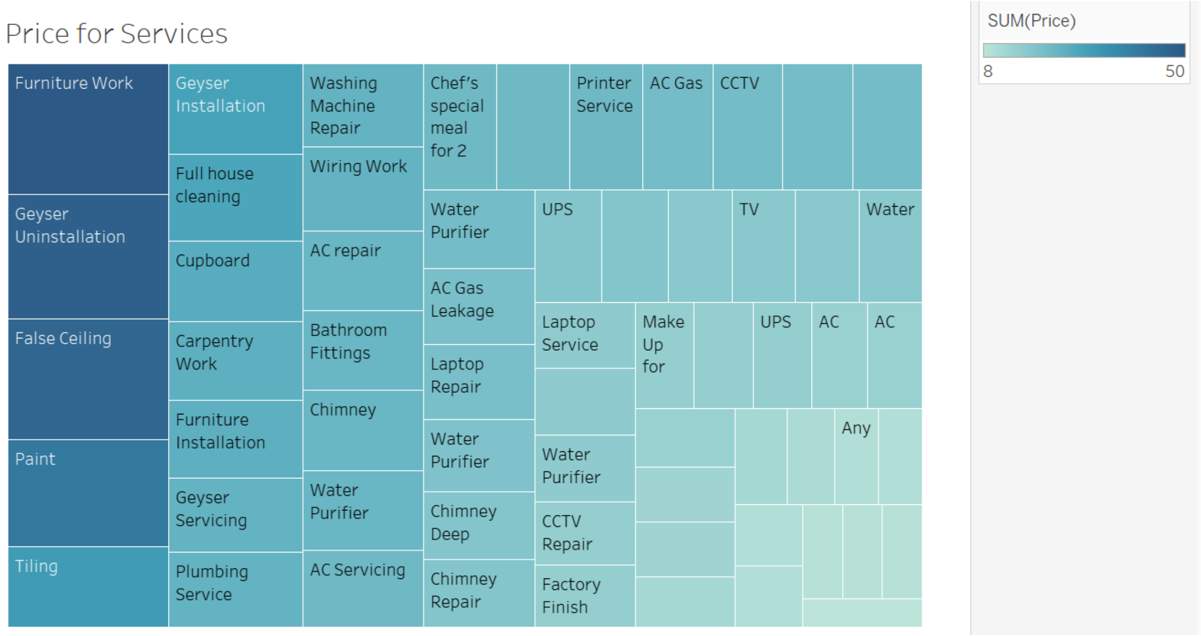


Visualisation 2: Distribution of payment methods



Count of Id for each Payment Method.

Visualisation 3: Pricing of different services


Price for Services
SUM(Price)
Furniture Work
Geyser Uninstallation
False Ceiling
Paint
Tiling
Geyser Installation
Full house cleaning
Cupboard
Carpentry Work
Furniture Installation
Geyser Servicing
Plumbing Service
Washing Machine Repair
Wiring Work
AC repair
Bathroom Fittings
Chimney
Water Purifier
AC Servicing
Chef's special meal for 2
Printer Service
AC Gas
CCTV
Water Purifier
AC Gas Leakage
Laptop Repair
Water Purifier
Chimney Deep
Chimney Repair
UPS
TV
Water
Laptop Service
Make Up for
UPS
AC
AC
Water Purifier
CCTV Repair
Factory Finish
Any