INTERNATIONALE HOCHSCHULE

DLBDSPDM01: - Building a Data Mart in SQL

Student : CEM OGUZ

Matriculation Number : 32008124

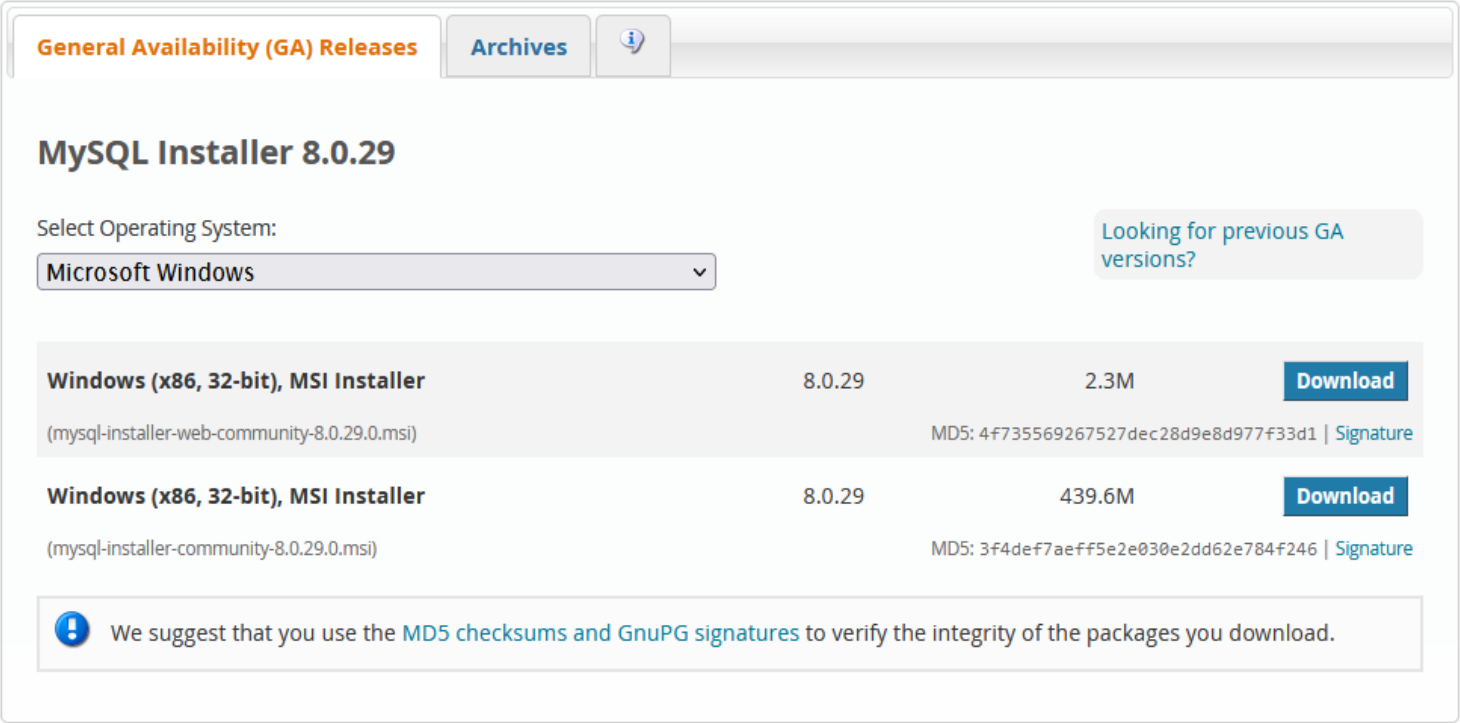
INTRODUCTION

- Installation of MySQL Server and MySQL WorkBench
- Creating Model and Tables
- Adding constraints & foreign keys
- Creating ER Diagram
- Setting Cardinalities & Configuration
- Creating pyhsical DB
- Inserting dummy data in tables
- Test Cases

Installation of Tools

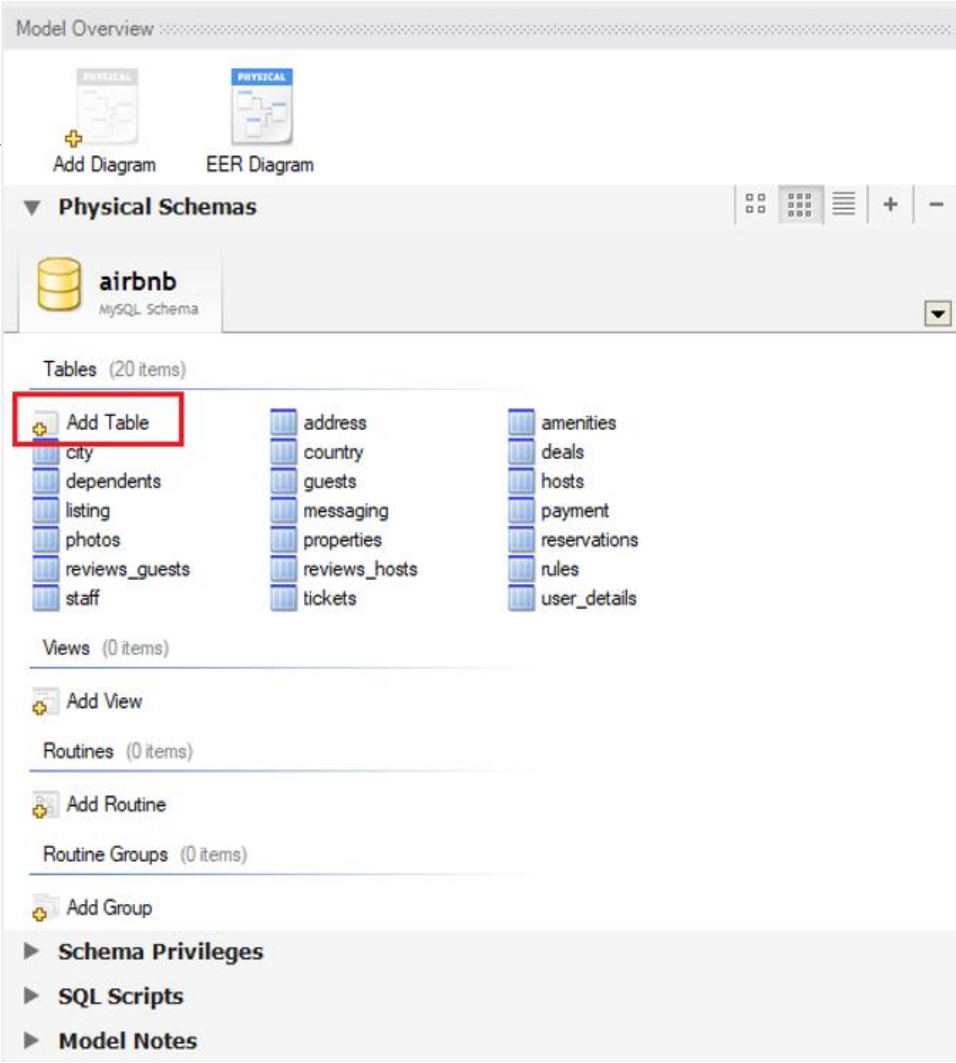
First step is to install MySQL server and a database design tool.

For this purpose, I chose MYSQL Installer and installed MySQL server and MySQL WorkBench in it.



Creating Model and Tables

- Once installation is done,
- Next step is to create a new model and new schema called «airbnb».
- Inside the model, I created 20 entities each of which will be a table in the database.
- For this purpose, I used «Add Table» icon and created tables as seen on the right hand side.



Adding Columns and Setting Data Types

- Once the tables are created, I added columns of each table, set their data types and column attributes like primary keys, not null, unsigned, auto increment etc. (where necessary)
- Below is the details of address table which represents other tables as an example.
- Other tables' details were configured in the same context.

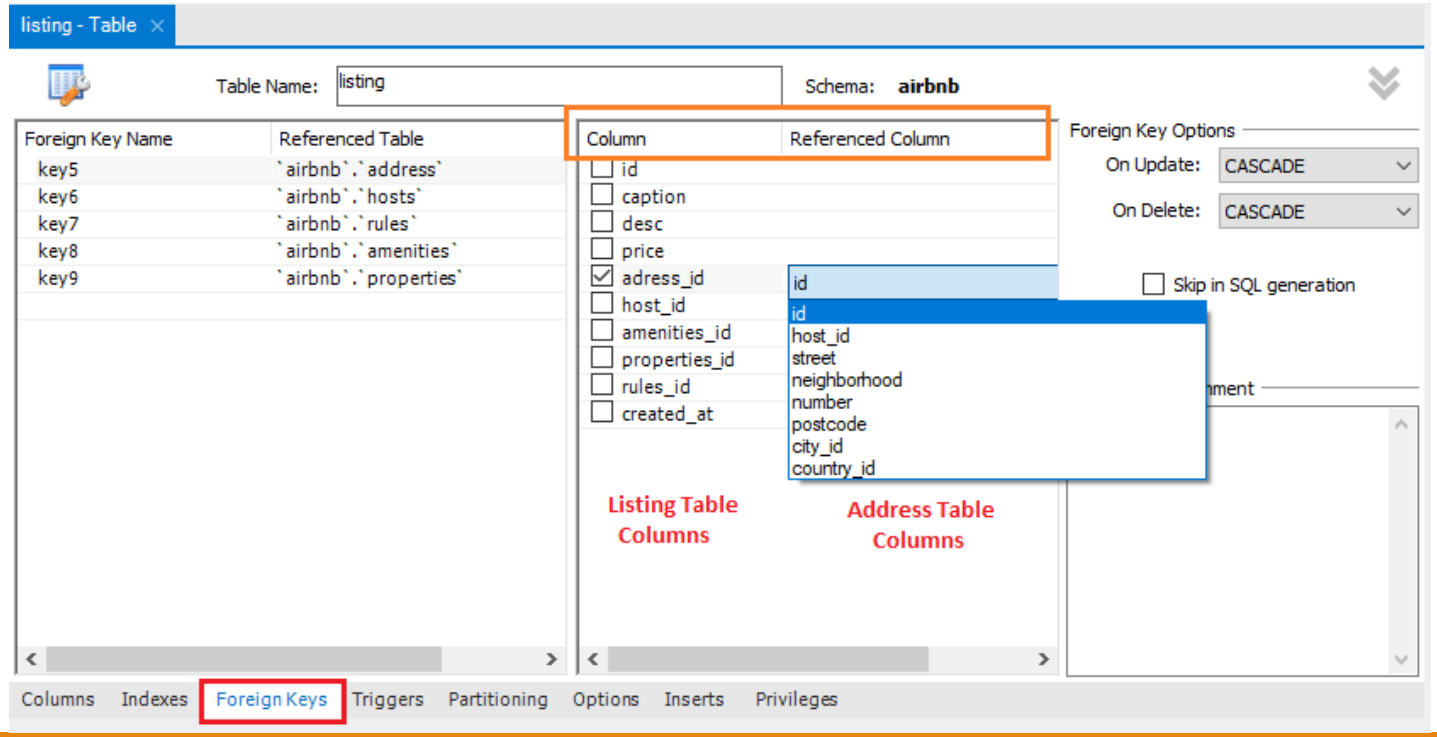
address - Table

Table Name: address Schema: **airbnb**

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT	☑	☑	☐	☐	☐	☐	☑	☐	
host_id	INT	☐	☑	☐	☐	☐	☐	☐	☐	
street	VARCHAR(20)	☐	☑	☐	☐	☐	☐	☐	☐	
neighborhood	VARCHAR(25)	☐	☑	☐	☐	☐	☐	☐	☐	
number	INT	☐	☑	☐	☐	☐	☐	☐	☐	
postcode	VARCHAR(10)	☐	☑	☐	☐	☐	☐	☐	☐	
city_id	INT	☐	☑	☐	☐	☐	☐	☐	☐	
country_id	INT	☐	☑	☐	☐	☐	☐	☐	☐	
		☐	☐	☐	☐	☐	☐	☐	☐	

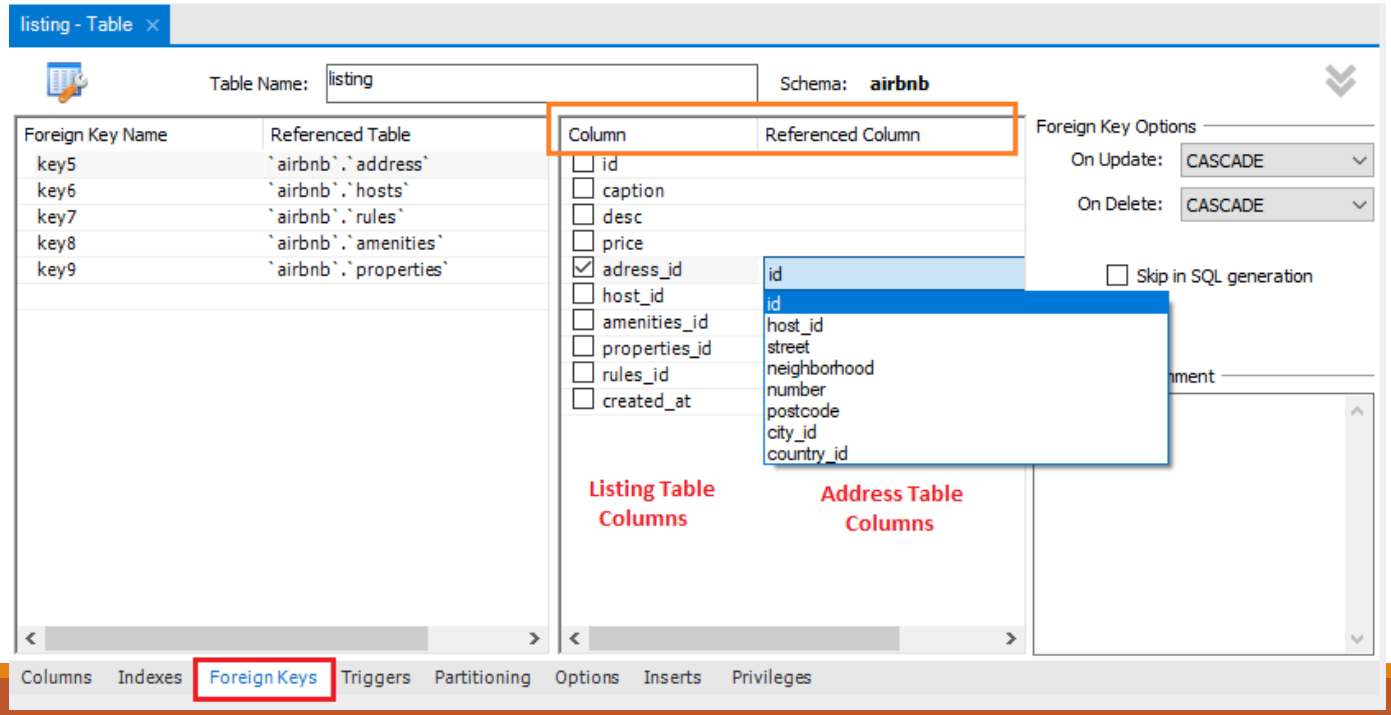
Setting Foreign Keys – (Listing Table Example)

- On the left half of the screen, I added foreign keys which reference to other tables.
- I named each one , and chose their referenced tables one by one from the right hand side.



Setting Foreign Keys – (Listing Table Example)

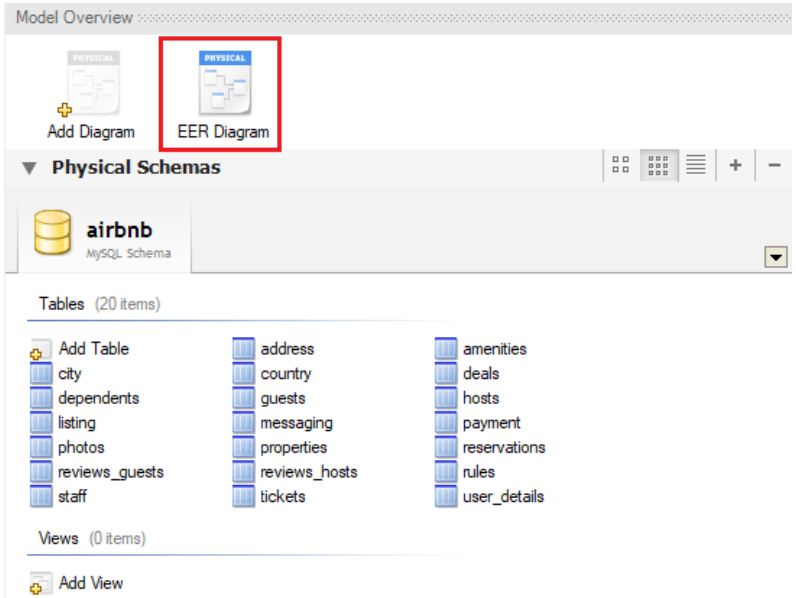
- On the right hand side of the screen is where I choose the column for each foreign key.
- Key5 foreign key, for example is chosen to be the **address_id** column.
- Since it references to Address table, I chose referenced column from address table, which is ID.
- So **adress_id on listing table is FK**, and it references to **ID (PK) on address table.**

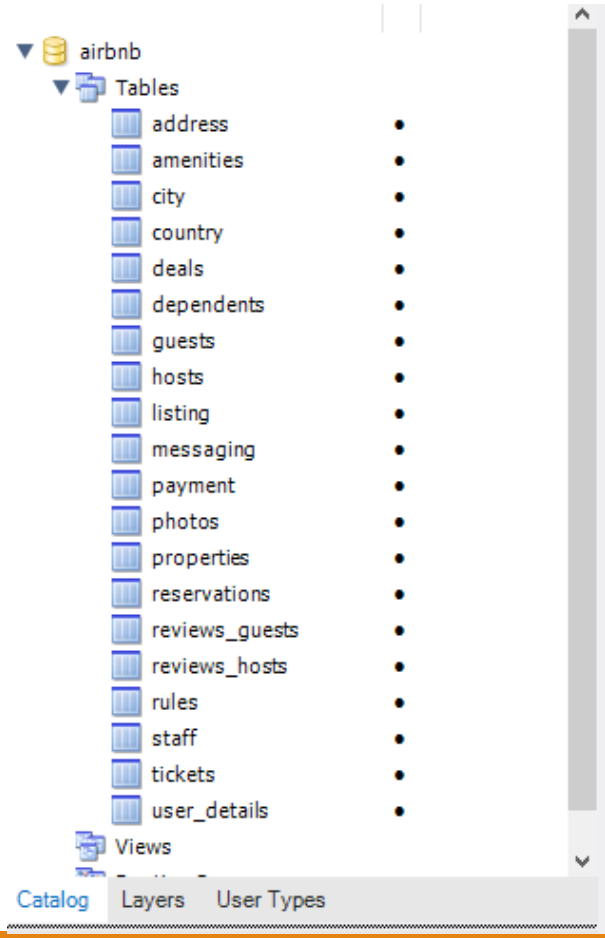


Creating ER Diagram

Next step is to create ER diagram. Clicking «Add Diagram» will do it.

In the new tab, I dragged and dropped all the tables into diagram space so that all the tables would be included prior to setting other relations.

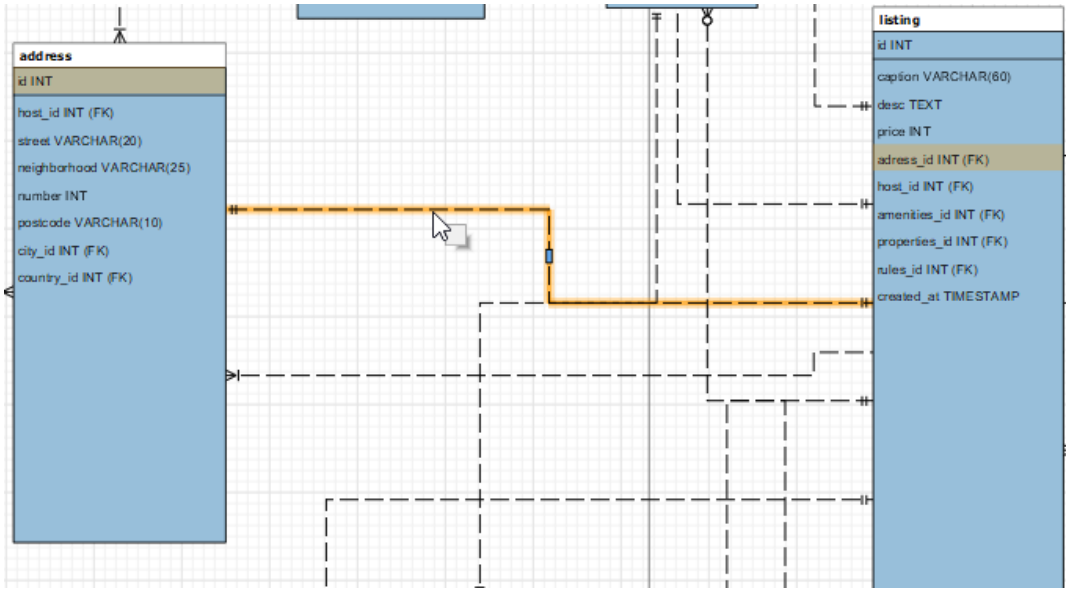


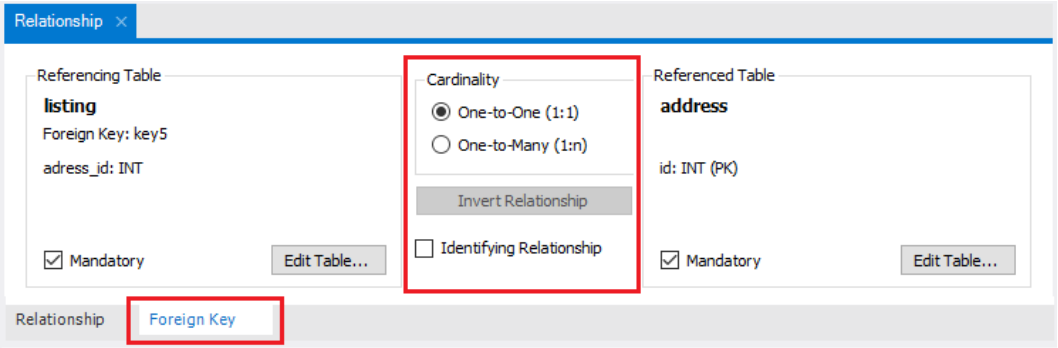


Configuring Cardinalities

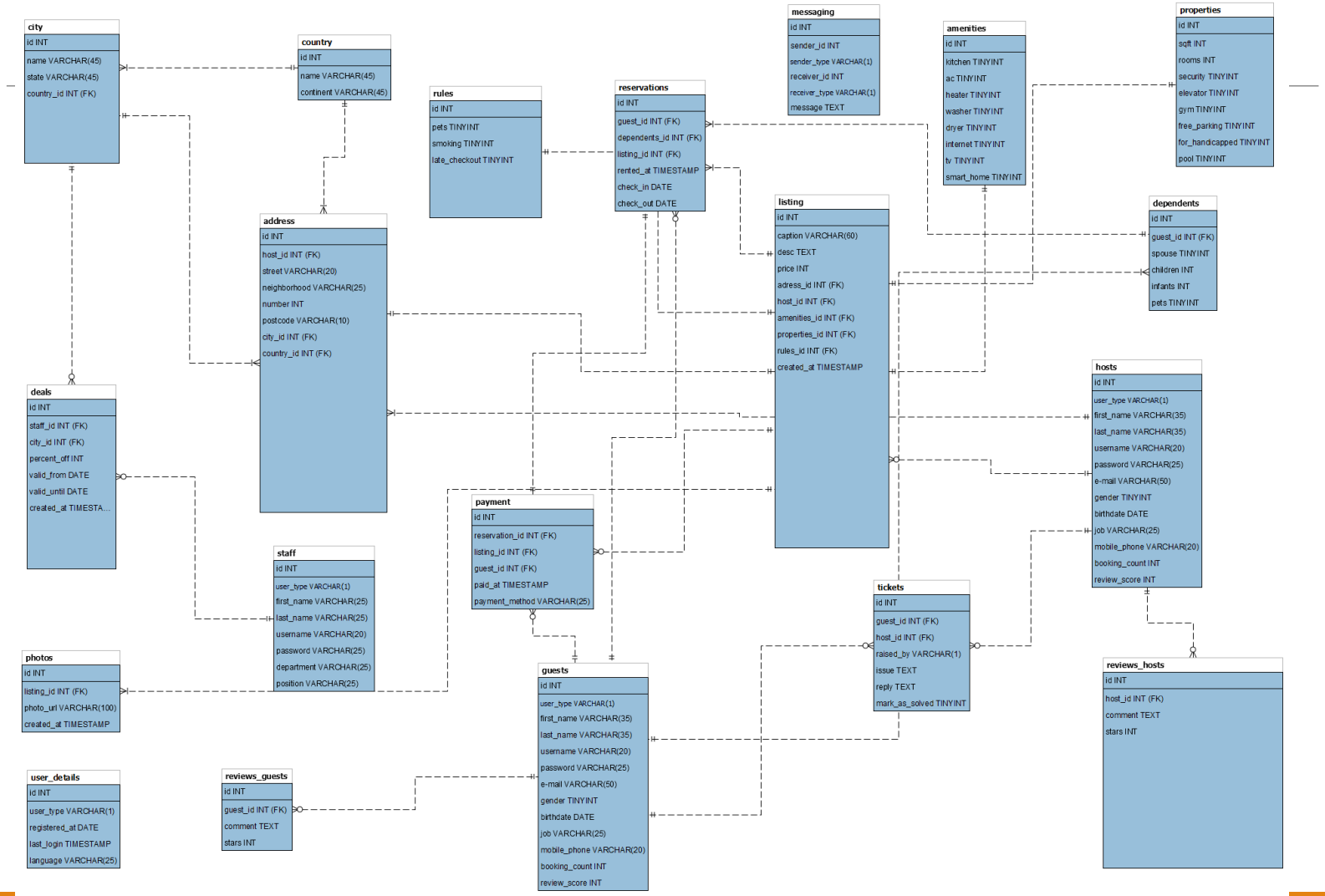
- After forming FK and PK relations, I came back to ER diagram to finetune cardinalities for each relation formed.

- By double clicking on each relation on EER Diagram, opening the relationship tab, I configured the cardinalities and other options for that relation so that they would come to final shapes.





Final EER Diagram



Pyhsical Creation of DB

Once all the tables, their data types, their relations, cardinalities and other options are fully configured, I can now generate the SQL code in order to physically create the database.

For this purpose, I used MySQL Forward Engineering Tool.

Upcoming slides will show each table's code one by one.

MySQL Forward Engineering Tool

Forward Engineer to Database

- Connection Options
- Options
- **Select Objects**
- Review SQL Script
- Commit Progress

Select Objects to Forward Engineer

To exclude objects of a specific type from the SQL Export, disable the corresponding checkbox. Press Show Filter and add objects or patterns to the ignore list to exclude them from the export.

☑ Export MySQL Table Objects
20 Total Objects, 20 Selected
Show Filter

☐ Export MySQL View Objects
0 Total Objects, 0 Selected
Show Filter

☐ Export MySQL Routine Objects
0 Total Objects, 0 Selected
Show Filter

☐ Export MySQL Trigger Objects
0 Total Objects, 0 Selected
Show Filter

☐ Export User Objects
0 Total Objects, 0 Selected
Show Filter

Pyhsical Creation of DB

```
-- -----------------------------------------------------
-- Table `airbnb`.`guests`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `airbnb`.`guests` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `user_type` VARCHAR(1) NOT NULL,
  `first_name` VARCHAR(35) NOT NULL,
  `last_name` VARCHAR(35) NOT NULL,
  `username` VARCHAR(20) NOT NULL,
  `password` VARCHAR(25) NOT NULL,
  `e-mail` VARCHAR(50) NOT NULL,
  `gender` TINYINT NOT NULL,
  `birthdate` DATE NOT NULL,
  `job` VARCHAR(25) NOT NULL,
  `mobile_phone` VARCHAR(20) NOT NULL,
  `booking_count` INT NULL,
  `review_score` INT NULL,
  PRIMARY KEY (`id`),
  UNIQUE INDEX `username_UNIQUE` (`username` ASC) VISIBLE,
  UNIQUE INDEX `e-mail_UNIQUE` (`e-mail` ASC) VISIBLE)
ENGINE = InnoDB;
```

```
CREATE TABLE IF NOT EXISTS `airbnb`.`dependents` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `guest_id` INT NOT NULL,
  `spouse` TINYINT NOT NULL,
  `children` INT NOT NULL,
  `infants` INT NOT NULL,
  `pets` TINYINT NOT NULL,
  PRIMARY KEY (`id`),
  INDEX `guest_id` (`guest_id` ASC) VISIBLE,
  CONSTRAINT `key4`
    FOREIGN KEY (`guest_id`)
    REFERENCES `airbnb`.`guests` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

```
CREATE TABLE IF NOT EXISTS `airbnb`.`photos` (
  `id` INT NOT NULL,
  `listing_id` INT NOT NULL,
  `photo_url` VARCHAR(100) NULL,
  `created_at` TIMESTAMP NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `key13`
    FOREIGN KEY (`listing_id`)
    REFERENCES `airbnb`.`listing` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

Pyhsical Creation of DB

```sql
CREATE TABLE IF NOT EXISTS `airbnb`.`hosts` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `user_type` VARCHAR(1) NOT NULL,
  `first_name` VARCHAR(35) NOT NULL,
  `last_name` VARCHAR(35) NOT NULL,
  `username` VARCHAR(20) NOT NULL,
  `password` VARCHAR(25) NOT NULL,
  `e-mail` VARCHAR(50) NOT NULL,
  `gender` TINYINT NOT NULL,
  `birthdate` DATE NOT NULL,
  `job` VARCHAR(25) NOT NULL,
  `mobile_phone` VARCHAR(20) NOT NULL,
  `booking_count` INT NULL,
  `review_score` INT NULL,
  PRIMARY KEY (`id`),
  UNIQUE INDEX `username_UNIQUE` (`username` ASC) VISIBLE,
  UNIQUE INDEX `e-mail_UNIQUE` (`e-mail` ASC) VISIBLE)
ENGINE = InnoDB;
```

```sql
-- -----------------------------------------------------
-- Table `airbnb`.`country`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `airbnb`.`country` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  `continent` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;
```

```sql
CREATE TABLE IF NOT EXISTS `airbnb`.`amenities` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `kitchen` TINYINT NOT NULL,
  `ac` TINYINT NOT NULL,
  `heater` TINYINT NOT NULL,
  `washer` TINYINT NOT NULL,
  `dryer` TINYINT NOT NULL,
  `internet` TINYINT NOT NULL,
  `tv` TINYINT NOT NULL,
  `smart_home` TINYINT NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;
```

Pyhsical Creation of DB

```sql
CREATE TABLE IF NOT EXISTS `airbnb`.`city` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  `state` VARCHAR(45) NOT NULL,
  `country_id` INT NOT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `key1`
    FOREIGN KEY (`country_id`)
    REFERENCES `airbnb`.`country` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

```sql
CREATE TABLE IF NOT EXISTS `airbnb`.`properties` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `sqft` INT NOT NULL,
  `rooms` INT NOT NULL,
  `security` TINYINT NOT NULL,
  `elevator` TINYINT NOT NULL,
  `gym` TINYINT NOT NULL,
  `free_parking` TINYINT NOT NULL,
  `for_handicapped` TINYINT NOT NULL,
  `pool` TINYINT NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;
```

```sql
CREATE TABLE IF NOT EXISTS `airbnb`.`address` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `host_id` INT NOT NULL,
  `street` VARCHAR(20) NOT NULL,
  `neighborhood` VARCHAR(25) NOT NULL,
  `number` INT NOT NULL,
  `postcode` VARCHAR(10) NOT NULL,
  `city_id` INT NOT NULL,
  `country_id` INT NOT NULL,
  PRIMARY KEY (`id`),
  INDEX `host_id` (`host_id` ASC) VISIBLE,
  INDEX `city_id` (`city_id` ASC) INVISIBLE,
  INDEX `country_id` (`country_id` ASC) VISIBLE,
  CONSTRAINT `fk_host_id`
    FOREIGN KEY (`host_id`)
    REFERENCES `airbnb`.`hosts` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `fk_city_id`
    FOREIGN KEY (`city_id`)
    REFERENCES `airbnb`.`city` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `fk_country_id`
    FOREIGN KEY (`country_id`)
    REFERENCES `airbnb`.`country` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

Pyhsical Creation of DB

```
-- -------------------------------------------------------
-- Table `airbnb`.`rules`
-- -------------------------------------------------------
CREATE TABLE IF NOT EXISTS `airbnb`.`rules` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `pets` TINYINT NOT NULL,
  `smoking` TINYINT NOT NULL,
  `late_checkout` TINYINT NOT NULL,
  PRIMARY KEY (`id`),
  INDEX `fk_rules_id` (`id` ASC) VISIBLE)
ENGINE = InnoDB;

-- -------------------------------------------------------
CREATE TABLE IF NOT EXISTS `airbnb`.`reviews_hosts` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `host_id` INT NOT NULL,
  `comment` TEXT NOT NULL,
  `stars` INT NOT NULL,
  PRIMARY KEY (`id`),
  INDEX `host_id` (`host_id` ASC) INVISIBLE,
  CONSTRAINT `key18`
    FOREIGN KEY (`host_id`)
    REFERENCES `airbnb`.`hosts` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

```
CREATE TABLE IF NOT EXISTS `airbnb`.`listing` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `caption` VARCHAR(60) NOT NULL,
  `desc` TEXT NOT NULL,
  `price` INT NOT NULL,
  `adress_id` INT NOT NULL,
  `host_id` INT NOT NULL,
  `amenities_id` INT NOT NULL,
  `properties_id` INT NOT NULL,
  `rules_id` INT NOT NULL,
  `created_at` TIMESTAMP NULL,
  PRIMARY KEY (`id`),
  INDEX `fk_amenities_id` (`amenities_id` ASC) INVISIBLE,
  INDEX `fk_properties_id` (`properties_id` ASC) INVISIBLE,
  INDEX `fk_rules_id` (`rules_id` ASC) VISIBLE,
  CONSTRAINT `key5`
    FOREIGN KEY (`adress_id`)
    REFERENCES `airbnb`.`address` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `key6`
    FOREIGN KEY (`host_id`)
    REFERENCES `airbnb`.`hosts` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `key7`
    FOREIGN KEY (`rules_id`)
    REFERENCES `airbnb`.`rules` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `key8`
    FOREIGN KEY (`amenities_id`)
    REFERENCES `airbnb`.`amenities` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `key9`
    FOREIGN KEY (`properties_id`)
    REFERENCES `airbnb`.`properties` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

Pyhsical Creation of DB

```
CREATE TABLE IF NOT EXISTS `airbnb`.`reservations` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `guest_id` INT NOT NULL,
  `dependents_id` INT NOT NULL,
  `listing_id` INT NOT NULL,
  `rented_at` TIMESTAMP NOT NULL,
  `check_in` DATE NOT NULL,
  `check_out` DATE NOT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `key14`
    FOREIGN KEY (`guest_id`)
    REFERENCES `airbnb`.`guests` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `key15`
    FOREIGN KEY (`dependents_id`)
    REFERENCES `airbnb`.`dependents` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `key16`
    FOREIGN KEY (`listing_id`)
    REFERENCES `airbnb`.`listing` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

```
CREATE TABLE IF NOT EXISTS `airbnb`.`staff` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `user_type` VARCHAR(1) NOT NULL,
  `first_name` VARCHAR(25) NOT NULL,
  `last_name` VARCHAR(25) NOT NULL,
  `username` VARCHAR(20) NOT NULL,
  `password` VARCHAR(25) NOT NULL,
  `department` VARCHAR(25) NOT NULL,
  `position` VARCHAR(25) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE INDEX `username_UNIQUE` (`username` ASC) VISIBLE)
ENGINE = InnoDB;
```

```
CREATE TABLE IF NOT EXISTS `airbnb`.`messaging` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `sender_id` INT NOT NULL,
  `sender_type` VARCHAR(1) NOT NULL,
  `receiver_id` INT NOT NULL,
  `receiver_type` VARCHAR(1) NOT NULL,
  `message` TEXT NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;
```

Pyhsical Creation of DB

```sql
CREATE TABLE IF NOT EXISTS `airbnb`.`payment` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `reservation_id` INT NOT NULL,
  `listing_id` INT NOT NULL,
  `guest_id` INT NOT NULL,
  `paid_at` TIMESTAMP NULL,
  `payment_method` VARCHAR(25) NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `key10`
    FOREIGN KEY (`reservation_id`)
    REFERENCES `airbnb`.`reservations` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `key11`
    FOREIGN KEY (`listing_id`)
    REFERENCES `airbnb`.`listing` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `key12`
    FOREIGN KEY (`guest_id`)
    REFERENCES `airbnb`.`guests` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

```sql
CREATE TABLE IF NOT EXISTS `airbnb`.`deals` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `staff_id` INT NOT NULL,
  `city_id` INT NOT NULL,
  `percent_off` INT NOT NULL,
  `valid_from` DATE NOT NULL,
  `valid_until` DATE NOT NULL,
  `created_at` TIMESTAMP NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `key2`
    FOREIGN KEY (`staff_id`)
    REFERENCES `airbnb`.`staff` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `key3`
    FOREIGN KEY (`city_id`)
    REFERENCES `airbnb`.`city` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

Pyhsical Creation of DB

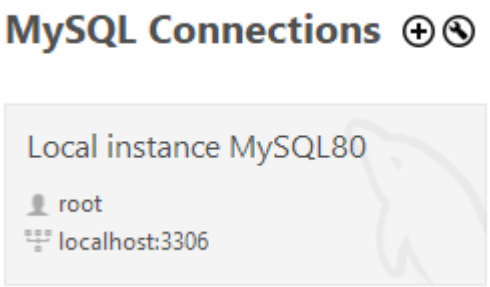
```sql
CREATE TABLE IF NOT EXISTS `airbnb`.`tickets` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `guest_id` INT NOT NULL,
  `host_id` INT NOT NULL,
  `raised_by` VARCHAR(1) NOT NULL,
  `issue` TEXT NOT NULL,
  `reply` TEXT NOT NULL,
  `mark_as_solved` TINYINT NOT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `key19`
    FOREIGN KEY (`guest_id`)
    REFERENCES `airbnb`.`guests` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `key20`
    FOREIGN KEY (`host_id`)
    REFERENCES `airbnb`.`hosts` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

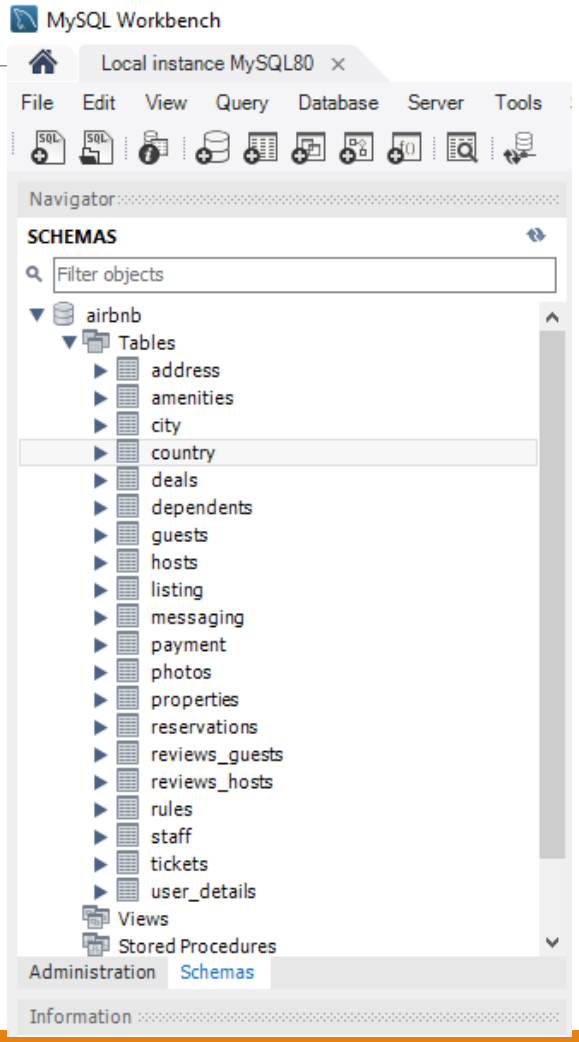
```sql
CREATE TABLE IF NOT EXISTS `airbnb`.`user_details` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `user_type` VARCHAR(1) NOT NULL,
  `registered_at` DATE NOT NULL,
  `last_login` TIMESTAMP NOT NULL,
  `language` VARCHAR(25) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;
```

```sql
CREATE TABLE IF NOT EXISTS `airbnb`.`reviews_guests` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `guest_id` INT NOT NULL,
  `comment` TEXT NOT NULL,
  `stars` INT NOT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `key17`
    FOREIGN KEY (`guest_id`)
    REFERENCES `airbnb`.`guests` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

Connection to DB

- After running the SQL code, the DB is created.
- So now, I can connect to server on my local machine and see the database.

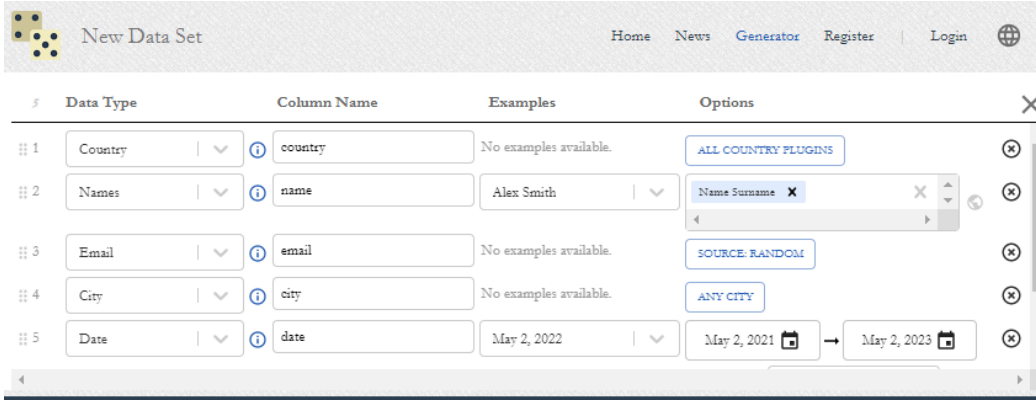




INSERTING DUMMY DATA

This is the first website I used for this purpose. : **generatedata.com**

It allows the user to choose the data type, table name, column names and finally creates SQL code along with dummy data.



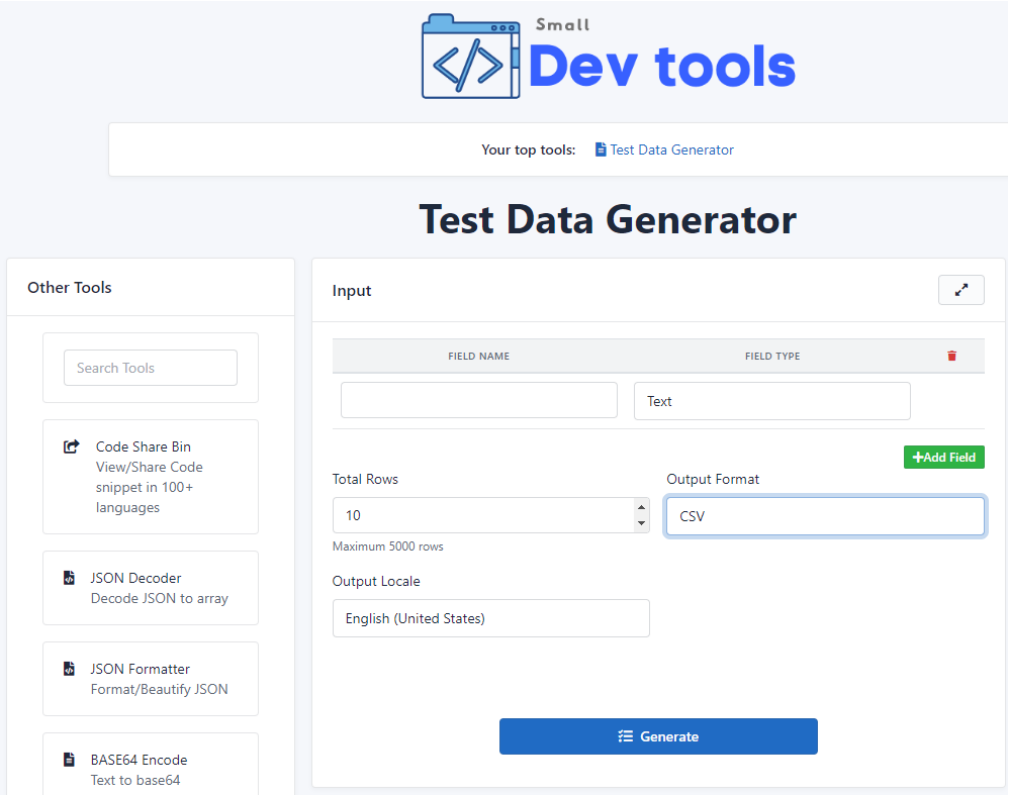
MYSQL

```mysql
DROP TABLE IF EXISTS `myTable`;

CREATE TABLE `myTable` (
  `id` mediumint(8) unsigned NOT NULL auto_increment,
  `country` varchar(100) default NULL,
  `name` varchar(255) default NULL,
  `email` varchar(255) default NULL,
  `city` varchar(255),
  `date` varchar(255),
  PRIMARY KEY (`id`)
) AUTO_INCREMENT=1;

INSERT INTO `myTable` (`country`,`name`,`email`,`city`,`date`)
VALUES
  ("Nigeria","Lesley Cook","velit.eget@aol.net","Emalahleni","Jun 14, 2022"),
  ("Germany","Velma Torres","aliquet.libero@yahoo.net","Farrukhabad-cum-Fatehgarh","Aug 9, 2022"),
  ("South Korea","Burke Cervantes","gravida.praesent@icloud.net","Belfast","Mar 23, 2022"),
  ("Spain","Mara Weeks","mollis.non.cursus@aol.edu","Voronezh","Sep 21, 2022"),
  ("Italy","Amena Cline","lorem.lorem.luctus@yahoo.edu","Springfield","May 7, 2021");
```

INSERTING DUMMY DATA

This is the second website I used for the same purpose : **smalldev.tools**

Collecting these two sources for dummy data, I inserted all of them into respective fields in the database.

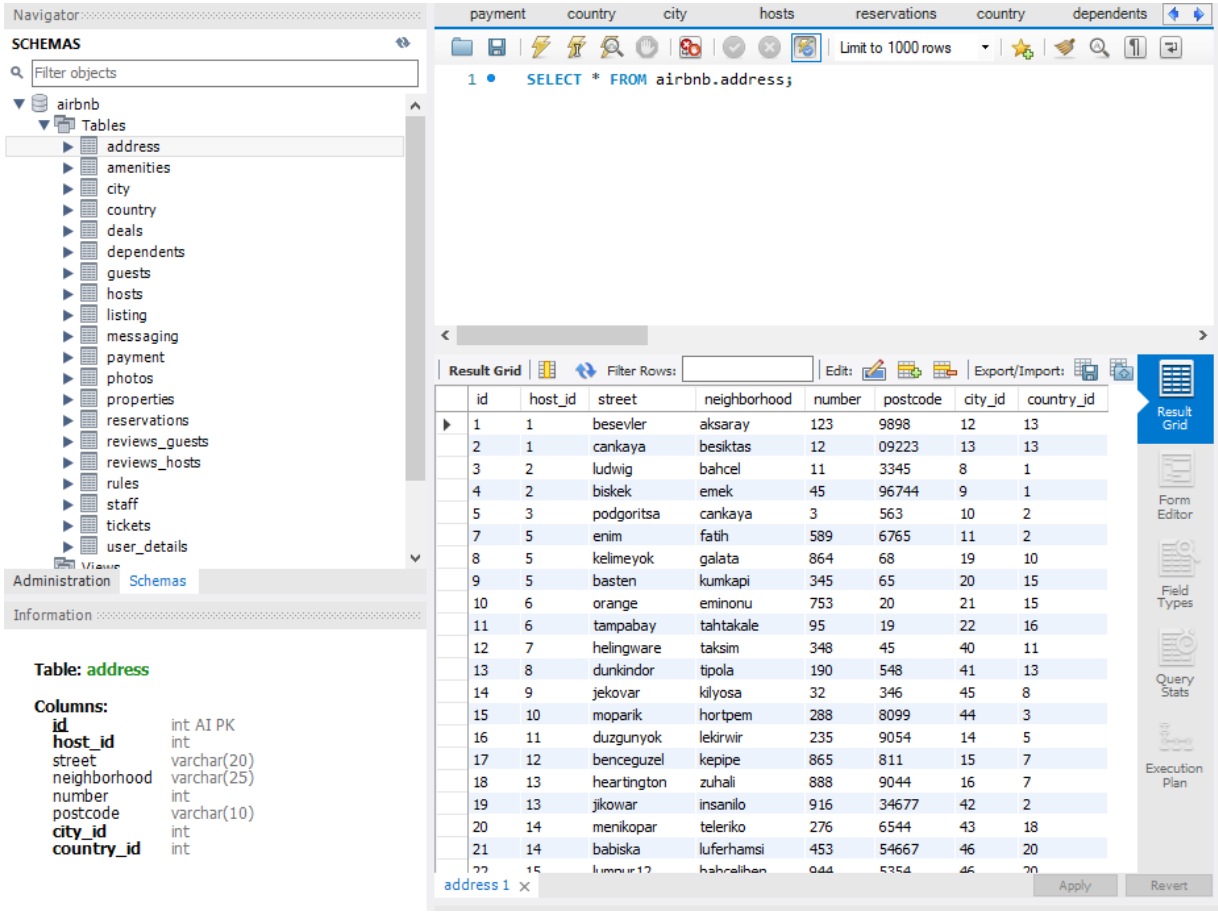


FINAL

Finally, I have a database with data in it which will allow me to do tests and run queries on.

On the right hand side is the address table for reference.

Each table has at least 20 rows of data.



TEST CASE 1

Get the name and last name of guests who has at least 1 infant as dependent

```
use airbnb;

SELECT dependents.infants as Infants,
guests.id as Guest_id,
guests.first_name as First_Name,
guests.last_name as Last_Name
FROM airbnb.guests, dependents
where dependents.infants >=1
and dependents.guest_id=guests.id;
```

Result Grid | Filter Rows: | Export:

Infants	Guest_id	First_Name	Last_Name
2	12	Ihsan	Aadams
4	4	Yoko	Reynolds
1	11	Cem	Love
1	16	Elif	Muzaffer
1	18	isimsiz	guzeltan
1	23	Atiba	Larin
1	25	Guti	Hernandez
1	28	Jack	Smith
1	29	Jackson	Oakwal
1	30	Hans	Peterov

TEST CASE 2

Get the Number of Reviews and Average Review Score for all Guests

```sql
use airbnb;

SELECT
guests.id AS Guest_ID,
CONCAT(guests.first_name, " ", guests.last_name) as Guest_Name_Surname,
count(guests.id) as Number_of_Reviews,
round(avg(reviews_guests.stars),1) as Review_Avg

from reviews_guests, guests
where guests.id=reviews_guests.guest_id
group by guests.id
order by guests.id asc;
```

Guest_ID	Guest_Name_Surname	Number_of_Reviews	Review_Avg
1	Urielle Ford	2	4.5
2	Ria Valencia	3	4.3
3	Giselle Cotton	1	5.0
4	Yoko Reynolds	1	3.0
5	Carter Lynn	2	3.0
7	Reagan Berk	2	3.0
8	Myles Magg	2	4.0
9	Gage Len	2	2.5
10	Onur Payne	2	4.5
11	Cem Love	2	4.5
12	Ihsan Aadams	2	5.0
13	Yusuf Pete	2	3.0
14	Isa Josh	2	3.5
15	Erdem Anil	2	3.5
16	Elif Muzaffer	2	2.0
17	zeynep akgoc	2	3.0
18	isimsiz guzeltan	3	3.3
19	buddy maltepe	2	4.5
20	mauris emek	2	3.0
21	kalantor insan	2	2.5
22	Joseph Durur	2	3.5
23	Atiba Larin	2	4.0
24	Meireles Cyle	2	3.5
25	Guti Hernandez	2	3.5
26	Joanne Gonzales	2	3.5
27	Natasha Peter	2	3.5
28	Jack Smith	2	1.0
29	Jackson Oakwal	2	4.0
30	Hans Peterov	2	2.5

TEST CASE 3

Get the reservations of 2020 with price higher than 150 Euros.

```
SELECT
reservations.listing_id as Listing_ID,
listing.host_id as Host_ID,
reservations.guest_id as Guest_ID,
price as Price,
Rented_at as Rented_at

FROM airbnb.listing
JOIN airbnb.reservations ON reservations.listing_id=airbnb.listing.id
Where price>150 and
(reservations.rented_at > '2020-01-01' and reservations.rented_at<'2020-12-31')
order by price;
```

Listing_ID	Host_ID	Guest_ID	Price	Rented_at
18	8	17	155	2020-10-18 10:10:10
22	13	21	156	2020-11-06 10:10:10
21	13	20	176	2020-10-26 10:10:10
15	11	14	178	2020-10-03 10:10:10
49	14	19	186	2020-07-21 10:10:10
30	19	30	194	2020-01-05 10:10:10
53	10	23	195	2020-08-27 10:10:10
14	10	13	205	2020-10-01 10:10:10
25	15	24	209	2020-11-27 10:10:10
20	12	19	212	2020-10-22 10:10:10
58	6	28	222	2020-08-14 10:10:10
54	10	24	235	2020-08-07 10:10:10
50	13	20	236	2020-07-23 10:10:10
57	7	27	243	2020-08-12 10:10:10
51	12	21	244	2020-07-24 10:10:10
56	8	26	245	2020-08-11 10:10:10
31	19	1	246	2020-01-25 10:10:10
55	9	25	276	2020-08-08 10:10:10
59	5	29	290	2020-08-15 10:10:10
52	11	22	304	2020-07-27 10:10:10
33	2	3	346	2020-02-05 10:10:10
34	3	4	355	2020-03-05 10:10:10

TEST CASE 3 CONT.

Get the reservations of 2021 with price higher than 150 Euros.

```
SELECT
reservations.listing_id as Listing_ID,
listing.host_id as Host_ID,
reservations.guest_id as Guest_ID,
price as Price,
Rented_at as Rented_at

FROM airbnb.listing
JOIN airbnb.reservations ON reservations.listing_id=airbnb.listing.id
Where price>150 and
(reservations.rented_at > '2021-01-01' and reservations.rented_at<'2021-12-31')
order by price;
```

Listing_ID	Host_ID	Guest_ID	Price	Rented_at
30	19	29	194	2021-04-17 10:10:10
28	18	27	231	2021-02-07 10:10:10
27	17	26	245	2021-01-07 10:10:10
31	19	30	246	2021-04-27 10:10:10
26	16	25	256	2021-01-04 10:10:10
33	2	2	346	2021-06-04 10:10:10
34	3	3	355	2021-06-08 10:10:10