**Restaurant table**

CREATE TABLE IF NOT EXISTS `restaurant` (

`Rest\_id` int(32) NOT NULL AUTO\_INCREMENT,

`Name` varchar(64) NOT NULL,

`Location\_id` int(32) NOT NULL,

`Cuisine` varchar(64) NOT NULL,

`Cost` int(32) NOT NULL,

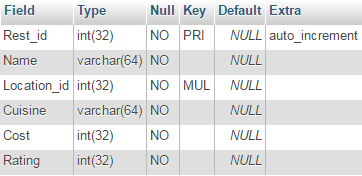
`Rating` int(32) NOT NULL,

PRIMARY KEY (`Rest\_id`),

KEY `Location\_id` (`Location\_id`)

);

desc restaurant;



**Customer Table**

CREATE TABLE IF NOT EXISTS `customer` (

`C\_id` int(32) NOT NULL AUTO\_INCREMENT,

`C\_name` varchar(64) NOT NULL,

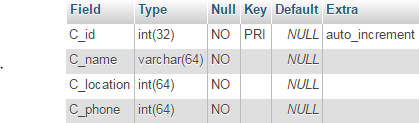
`C\_location` int(64) NOT NULL,

`C\_phone` int(64) NOT NULL,

PRIMARY KEY (`C\_id`)

);

desc customer;



**Location Table**

CREATE TABLE IF NOT EXISTS `location` (

`Area` varchar(1024) NOT NULL,

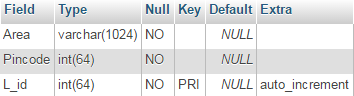
`Pincode` int(64) NOT NULL,

`L\_id` int(64) NOT NULL AUTO\_INCREMENT,

PRIMARY KEY (`L\_id`)

);

desc location;



**Menu Table**

CREATE TABLE IF NOT EXISTS `menu` (

`Rest\_id` int(32) NOT NULL,

`Food` varchar(64) NOT NULL,

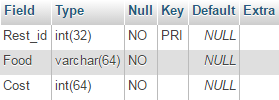
`Cost` int(64) NOT NULL,

PRIMARY KEY (`Rest\_id`),

KEY `Rest\_id` (`Rest\_id`)

);

desc menu;



**Orders Table**

CREATE TABLE IF NOT EXISTS `orders` (

`O\_id` int(32) NOT NULL AUTO\_INCREMENT,

`Customer\_id` int(32) NOT NULL,

`Restaurant\_id` int(32) NOT NULL,

`Bill` int(32) NOT NULL,

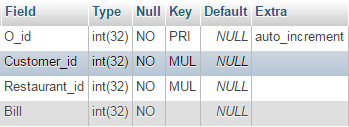
PRIMARY KEY (`O\_id`),

KEY `Customer\_id` (`Customer\_id`),

KEY `Restaurant\_id` (`Restaurant\_id`)

);

desc orders;



-- Constraints for table `menu`

ALTER TABLE `menu`

ADD CONSTRAINT `menu\_ibfk\_1` FOREIGN KEY (`Rest\_id`) REFERENCES `restaurant` (`Rest\_id`) ON DELETE CASCADE ON UPDATE CASCADE;

-- Constraints for table `orders`

ALTER TABLE `orders`

ADD CONSTRAINT `orders\_ibfk\_2` FOREIGN KEY (`Restaurant\_id`) REFERENCES `restaurant` (`Rest\_id`) ON DELETE CASCADE ON UPDATE CASCADE,

ADD CONSTRAINT `orders\_ibfk\_1` FOREIGN KEY (`Customer\_id`) REFERENCES `customer` (`C\_id`) ON DELETE CASCADE ON UPDATE CASCADE;

-- Constraints for table `restaurant`

ALTER TABLE `restaurant`

ADD CONSTRAINT `restaurant\_ibfk\_1` FOREIGN KEY (`Location\_id`) REFERENCES `location` (`L\_id`) ON DELETE CASCADE ON UPDATE CASCADE;

**Relationship Schema**

Restaurant

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Rest\_id | Name | Location\_id | Cuisine | Cost | Rating |

Location

|  |  |  |
| --- | --- | --- |
| Area | Pincode | L\_id |

Customer

|  |  |  |  |
| --- | --- | --- | --- |
| C\_id | C\_name | C\_location | C\_phone |

Orders

|  |  |  |  |
| --- | --- | --- | --- |
| O\_id | Customer\_id | Restaurant\_id | Bill |

Menu

|  |  |  |
| --- | --- | --- |
| Rest\_id | Food | Cost |

Horizontal Partitioning

**Range Partitioning**

CREATE TABLE IF NOT EXISTS `restaurant2` (

`Rest\_id` int(32) NOT NULL AUTO\_INCREMENT,

`Name` varchar(64) NOT NULL,

`Location\_id` int(32) NOT NULL,

`Cuisine` varchar(64) NOT NULL,

`Cost` int(32) NOT NULL,

`Rating` int(32) NOT NULL,

PRIMARY KEY (`Rest\_id`,`Cost`),

KEY `Location\_id` (`Location\_id`)

)

PARTITION BY RANGE (Cost) (

PARTITION L5 VALUES LESS THAN (500),

PARTITION L10 VALUES LESS THAN (1000),

PARTITION G10 VALUES LESS THAN MAXVALUE

);

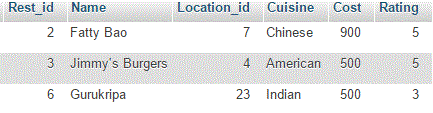
SELECT \* FROM `restaurant2`;



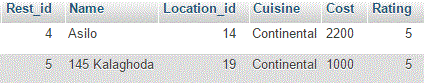
SELECT \* FROM ‘restaurant2’ PARTITION (L5);

ddb1.GIF

SELECT \* FROM ‘restaurant2’ PARTITION (L10);



SELECT \* FROM ‘restaurant2’ PARTITION (G10);



**List Partitioning**

CREATE TABLE IF NOT EXISTS `location` (

`Area` varchar(1024) NOT NULL,

`Pincode` int(64) NOT NULL,

`L\_id` int(64) NOT NULL AUTO\_INCREMENT,

PRIMARY KEY (`L\_id`)

)

PARTITION BY LIST (L\_id) (

PARTITION north VALUES IN (1,2,3,4,5,6),

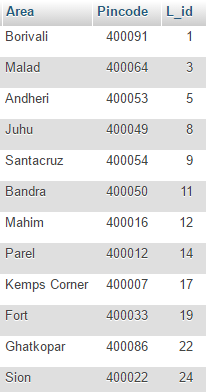
PARTITION burbs VALUES IN (7,8,9,10,11),

PARTITION south VALUES IN (12,13,14,15,16,17,18,19,20),

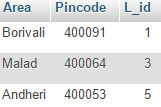
PARTITION central VALUES IN (21,22,23,24,25)

);

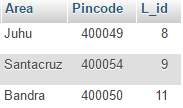
SELECT \* FROM `location`;



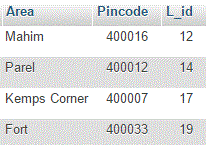
SELECT \* FROM `location` PARTITION (north);



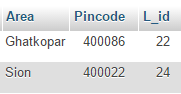
SELECT \* FROM `location` PARTITION (burbs);



SELECT \* FROM `location` PARTITION (south);



SELECT \* FROM `location` PARTITION (central);



**Key Partitioning**

CREATE TABLE IF NOT EXISTS `location` (

`Area` varchar(1024) NOT NULL,

`Pincode` int(64) NOT NULL,

`L\_id` int(64) NOT NULL AUTO\_INCREMENT,

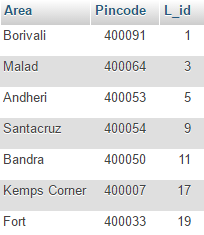
PRIMARY KEY (`L\_id`)

)

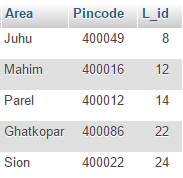
PARTITION BY KEY()

PARTITIONS 2;

SELECT \* FROM `location1` PARTITION (p0);



SELECT \* FROM `location1` PARTITION (p1);



**Hash Partitioning**

CREATE TABLE IF NOT EXISTS `location3` (

`Area` varchar(1024) NOT NULL,

`Pincode` int(64) NOT NULL,

`L\_id` int(64) NOT NULL AUTO\_INCREMENT,

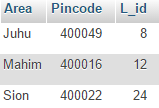
PRIMARY KEY (`L\_id`)

)

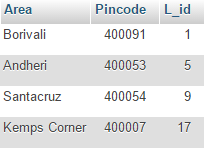
PARTITION BY HASH(L\_id)

PARTITIONS 4;

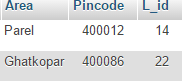
SELECT \* FROM `location3` PARTITION (p0);



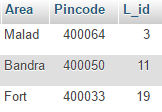
SELECT \* FROM `location3` PARTITION (p1);



SELECT \* FROM `location3` PARTITION (p2);



SELECT \* FROM `location3` PARTITION (p3);



Vertical Fragmentation

**Vertical Partitioning**

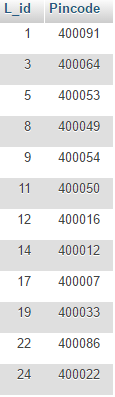
CREATE VIEW v1 AS

SELECT L\_id, Pincode FROM location;

CREATE VIEW v2 AS

SELECT L\_id, Area FROM location

SELECT \* FROM `v1`;



SELECT \* FROM `v2`;

