Database Specifications

Next-Gen Restaurant Application

Parv Bhatt (pnb5078@psu.edu)
Raghava Rao Sunkanapally (rxs6097@psu.edu)

School of Graduate Professional Studies
Information Science Department

IN SC 521 - Introduction to Database Concepts

October, 2021

DOCUMENT CONTROL

Work carried out by:

Name	Email Address	Other
Parv Bhatt	pnb5078@gmail.com	
Raghava Rao Sunkanapally	rxs6097@gmail.com	

Revision Sheet

Release No.	Date	Revision Description
1	10/31/2021	Gathered the data requirements from a software specifications software
2	11/04/2021	Updated the data requirements table as suggested by Dr. Barb
3	11/08/2021	Built an ER Diagram and wrote about Assumptions/Constraints
4	11/15/2021	Built a Logical Design of the ER Diagram
5	11/22/2021	Performed Normalization on the tables
6	12/05/2021	Implemented the physical design in SQL

DATABASE SPECIFICATIONS

TABLE OF CONTENTS

MILESTONE 1: DATA REQUIREMENTS

System Name or Title

Next – Gen Restaurant Application for Gotham City University

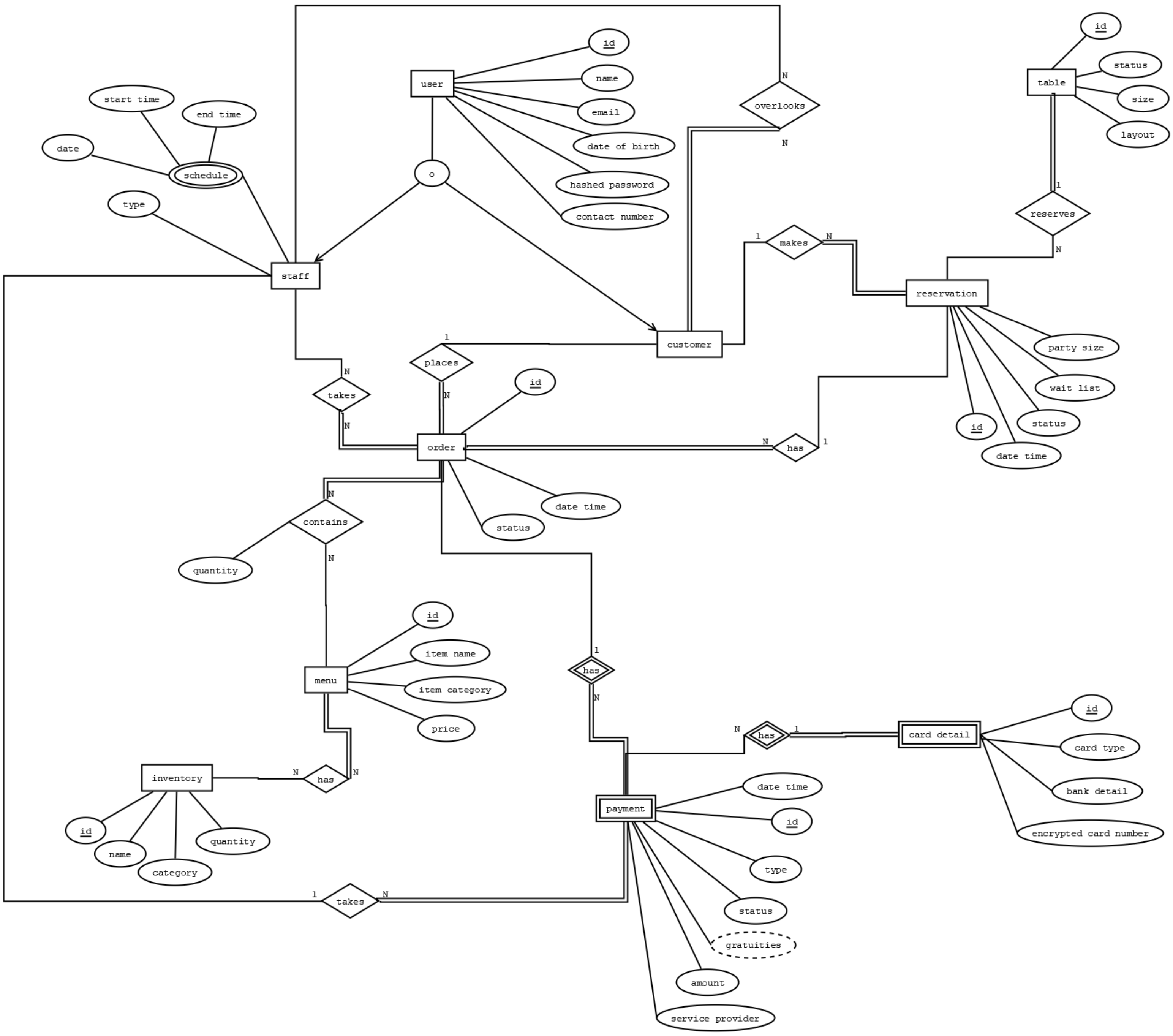
Core Requirements

No.	Requirement	Referenced page in SRS	Referenced Section in SRS	Referenced Paragraph in Section
DR1	The database should store information about customers, like name, id, contact number, dob	3 6 6 11	1.2 2.3 2.3 3.5	1 2 3 3.5.3.3 – 3.5.3.4
DR2	The database should store information about orders.	3	1.2	Point 6
DR3	The database should store information about table layout.	3 5	1.2 2.2	Point 1 Bullet 1
DR4	The database should store party size, reservation time/date, table when assigned. The database should also store the information about table occupancy and status. A waiting list number should also be maintained.	5 5 12 12 10 11 11 8	2.1 2.2 3.5 3.6 3 3 3 3	2 Bullet 7 3.5.6 3.6.2 3.5.3.2 3.5.3.7 3.5.3.8 3.1.1
DR5	The database should store the information about menu items and ingredients	4 7	1.2 2.3	Point 9 Table 1
DR6	The database should store the information about staff details and scheduling.	5	2.1	2
DR7	The database should store the information about users' roles, dob, authenticated usernames, and hashed password. User's roles include management members, servers, host/hostess, kitchen staff	13 13 6 8 10	5.1 5.2 2.3 3 3	5.1.1 5.2.1 2 3.1.2 3.5.3.1

DR8	The database should store the encrypted information about customers payment card details and bank account details.	13 13	5.3 5.3	5.3.1 5.3.2
DR9	The database should store the information about employee gratuities paid.	9	3.5	3.5.1.5
DR10	The database should store the information about customer payments from external third-party payment processing services and payment status	8	2.5	2.5.1
DR11	The database should store information about bar order/bar tab.	10	3	3.5.1.10

MILESTONE 2: CONCEPTUAL DESIGN

ER Diagram

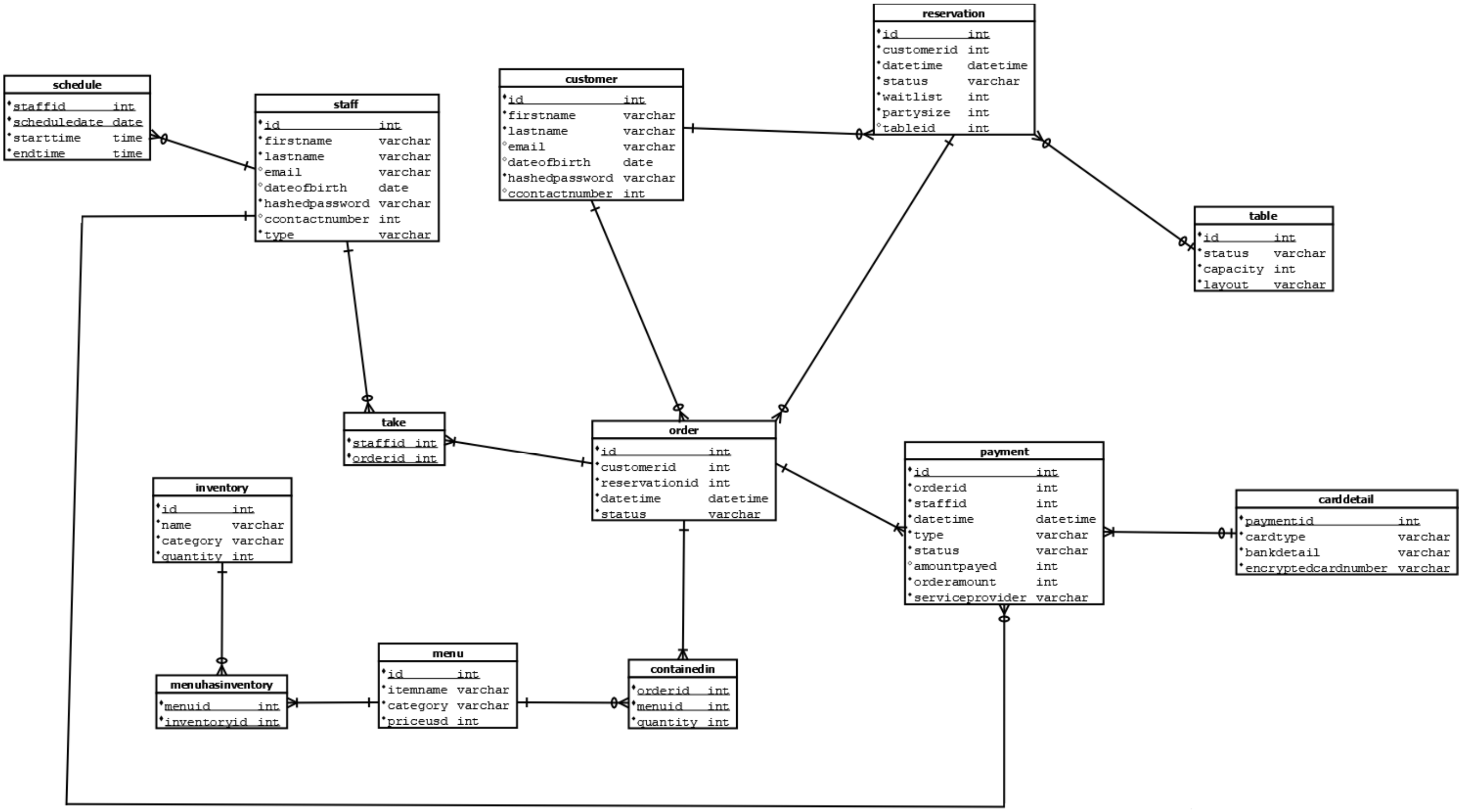


Assumptions and Constraints

A: Customers visit this restaurant.
A: Customers can reserve their own table. If all tables are occupied, he/she will be added to the wait queue.
A: Staff member can be a customer also.
A: Each server (staff) takes order placed by the customer.
A: Kitchen staff manages the inventory.
A: A particular customer is managed by a single staff member.
A: The gratuities associated with a customer/order is linked to the corresponding staff member.
A: Payment stores total order amount and gratuities amount.
A: For each non-cash payment done by customer, encrypted card detail is stored.
A: Each card detail is identified using weak entity; set "card details" discriminator key id.
C: Menu will have only those items which can be prepared from the current inventory.
C: Alcoholic beverage will only be served to customers whose age is greater than 21 years.
C: Alcoholic beverage will be served only by staff whose age is more than 21 years.

MILESTONE 3: LOGICAL DESIGN

Entity Relationship Diagram



Entity name: staff

Attributes:

id, firstname, lastname, email, dateofbirth, hashedpassword, contactnumber, type, schedule

Functional dependencies:

id → firstname, lastname, email, dateofbirth, hashedpassword, contactnumber, type, schedule

Attributes not in FD	Attributes on the left	Attributes on both sides	Attributes on the right side
	id		firstname, lastname, email, dateofbirth, hashedpassword, type, schedule, contactnumber

Attribute closures (if any):

id+ = firstname, lastname, email, dateofbirth, hashedpassword, contactnumber, type, schedule

Unique keys: the key for this table is/are

id

Entity name: customer

Attributes:

id, firstname, lastname, email, dateofbirth, hashedpassword, contactnumber

Functional dependencies:

id → firstname, lastname, email, dateofbirth, hashedpassword, contactnumber

Attributes not in FD	Attributes on the left	Attributes on both sides	Attributes on the right side
	id		firstname, lastname, email, dateofbirth, hashedpassword, contactnumber

Attribute closures (if any):

id+ = firstname, lastname, email, dateofbirth, hashedpassword, contactnumber

Unique keys: the key for this table is/are

id

Entity name: reservation

Attributes:

id, customerid, datetime, status, waitlist, partysize, tableid

Functional dependencies:

id → datetime, status, waitlist, partysize
customerid → customer
tableid → table

Attributes not in FD	Attributes on the left	Attributes on both sides	Attributes on the right side
	id		datetime, status, waitlist, partysize
	customerid		customer
	tableid		table

Attribute closures (if any):
id+ = customerid, tableid, datetime, status, waitlist, partysize
customerid+ = customer
tableid+ = table

Unique keys: the key for this table is/are
id

Entity name: table

Attributes:

id, status, capacity, layout

Functional dependencies:

id → status, capacity, layout

Attributes not in FD	Attributes on the left	Attributes on both sides	Attributes on the right side
	id		status, capacity, layout

Attribute closures (if any):
id+ = status, capacity, layout

Unique keys: the key for this table is/are
id

Entity name: order

Attributes:

id, status, customerid, datetime, reservationid

Functional dependencies:

id → status, datetime, customerid, reservationid
customerid → customer
reservationid → reservation

Attributes not in FD	Attributes on the left	Attributes on both sides	Attributes on the right side
	id		status, datetime
	customerid		customer
	reservationid		reservationid

Attribute closures (if any):
id+ = status, datetime
customerid+ = customer
reservationid+ = reservation

Unique keys: the key for this table is/are
id

Entity name: payment

Attributes:

id, orderid, staffid, datetime, type, status, amountpaid, orderamount, serviceprovider

Functional dependencies:

id → datetime, type, status, amountpaid, orderamount, serviceprovider, orderid, staffid
orderid → order
staffid → staff

Attributes not in FD	Attributes on the left	Attributes on both sides	Attributes on the right side
	id		datetime, type, status, amountpaid, orderamount, serviceprovider
	orderid		order
	staffid		staff

Attribute closures (if any):

id+ = status, datetime, type, amountpaid, orderamount, serviceprovider, orderid, staffid
orderid+ = order
staffid+ = staff

Unique keys: the key for this table is/are

id

Entity name: carddetail

Attributes:

paymentid, cardtype. bankdetail, encryptedcardnumber

Functional dependencies:

paymentid → payment, cardtype. bankdetail, encryptedcardnumber

Attributes not in FD	Attributes on the left	Attributes on both sides	Attributes on the right side
	paymentid		payment, cardtype. bankdetail, encryptedcardnumber

Attribute closures (if any):

paymentid+ = payment, cardtype, bankdetail, encryptedcardnumber

Unique keys: the key for this table is/are

paymentid

Entity name: menu

Attributes:

id, itemname, category, priceusd, quantity, orderid

Functional dependencies:

id → itemname, category, priceusd

Attributes not in FD	Attributes on the left	Attributes on both sides	Attributes on the right side
	id		itemname, category, priceusd

Attribute closures (if any):

id+ = itemname, category, priceusd

Unique keys: the key for this table is/are

id

Entity name: inventory

Attributes:

id, name, category, quantity, menuid

Functional dependencies:

id → name, category, quantity

Attributes not in FD	Attributes on the left	Attributes on both sides	Attributes on the right side
	id		name, category, quantity

Attribute closures (if any):
id+ = name, category, quantity

Unique keys: the key for this table is/are
id

Assumptions and Constraints

A: Customers visit this restaurant.
A: Customers can reserve their own table. If all tables are occupied, he/she will be added to the wait queue.
A: Staff member can be a customer also.
A: Each server (staff) takes order placed by the customer.
A: Kitchen staff manages the inventory.
A: A particular customer is managed by a single staff member.
A: The gratuities associated with a customer/order is linked to the corresponding staff member.
A: Payment stores total order amount and gratuities amount.
A: For each non-cash payment done by customer, encrypted card detail is stored.
A: Each card detail is identified using weak entity; set "card details" discriminator key id.
C: Menu will have only those items which can be prepared from the current inventory.
C: Alcoholic beverage will only be served to customers whose age is greater than 21 years.
C: Alcoholic beverage will be served only by staff whose age is more than 21 years.

MILESTONE 4: NORMALIZATION AND

MILESTONE 5: PHYSICAL DESIGN

Assumptions and Constraints

A: Customers visit this restaurant.
A: Customers can reserve their own table. If all tables are occupied, he/she will be added to the wait queue.
A: Staff member can be a customer also.
A: Each server (staff) takes order placed by the customer.
A: Kitchen staff manages the inventory.
A: A particular customer is managed by a single staff member.
A: The gratuities associated with a customer/order is linked to the corresponding staff member.
A: Payment stores total order amount and gratuities amount.
A: For each non-cash payment done by customer, encrypted card detail is stored.
A: Each card detail is identified using weak entity; set "card details" discriminator key id.
C: Menu will have only those items which can be prepared from the current inventory.
C: Alcoholic beverage will only be served to customers whose age is greater than 21 years.
C: Alcoholic beverage will be served only by staff whose age is more than 21 years.

Tables

Name of the table	***customer***			
Description	a customer is a person that visits the restaurant.			
Attribute	**Description**	**Type**	**Examples of values**	**Notes**
id	ID of a customer	number	12345	Unique, Not Null
firstname	First name of the customer	varchar	Raghava	Not Null
lastname	Last name of the customer	varchar	Sunkanapally	Can be null
email	Email of the customer	varchar	abc@gmail.com	Unique, Can be Null
dateofbirth	Date of Birth of the customer	date	01/01/1995	Can be Null
hashedpassword	Encrypted hashed password for customer login	varchar	ABC123@#	Not Null
contactnumber	Contact Number of the customer	number	9999999999	Unique, Can be Null
Functional Dependencies and Keys				
Functional dependencies	id → firstname, lastname, email, dateofbirth, hashedpassword, contactnumber			

Candidate keys	**id**	
Normalization		
1NF	**Yes**	All cells contain a unique value
2NF	**Yes**	Table is in 1NF and no prime attribute is dependent on any proper subset of any candidate key of the table
3NF	**Yes**	All the non-key attributes depend only on a key
BCNF	**Yes**	Every functional dependency X → Y, X is the super key of the table
Physical Design		
Primary Key	id	
Foreign Keys	-	
SQL Code	CREATE TABLE customer (id NUMBER(10) PRIMARY KEY, firstname VARCHAR2(50) NOT NULL, lastname VARCHAR2(50), email VARCHAR2(255) UNIQUE, dateofbirth DATE, hashedpassword VARCHAR2(255) NOT NULL, contactnumber NUMBER(20) UNIQUE);	
Count of records in the table	**50**	

…

Name of the table	***staff***			
Description	a staff is a person that works for our restaurant. There are many types of staff: Chef, Waiter, Management, Owner, etc.			
Attribute	**Description**	**Type**	**Examples of values**	**Notes**
id	ID of a staff	number	123	Unique, Not Null
firstname	First Name of Staff	varchar	Parv	Not Null
lastname	Last Name of Staff	varchar	Bhatt	Not Null
email	Email of Staff	varchar	abc@gmail.com	Unique, Not null
dateofbirth	date of birth of staff	date	01/01/1997	Not Null
hashedpassword	hashed passwords of the staff	varchar	abcdefgh	Not Null
contactnumber	Contact number of a staff member	number	9999999999	Not Null
type	Designation of the staff	varchar	host, server, chef	Not Null
Functional Dependencies and Keys				

Functional dependencies	id → firstname, lastname, email, dateofbirth, hashedpassword, contactnumber, type	
Candidate keys	**id**	
Normalization		
1NF	**Yes**	All cells contain a unique value
2NF	**Yes**	Table is in 1NF and no prime attribute is dependent on any proper subset of any candidate key of the table
3NF	**Yes**	All the non-key attributes depend only on a key
BCNF	**Yes**	Every functional dependency X → Y, X is the super key of the table
Physical Design		
Primary Key	id	
Foreign Keys	-	
SQL Code	CREATE TABLE staff (id NUMBER(10) PRIMARY KEY, firstname VARCHAR2(50) NOT NULL, lastname VARCHAR2(50) NOT NULL, email VARCHAR2(255) UNIQUE NOT NULL, dateofbirth DATE NOT NULL, hashedpassword VARCHAR2(255) NOT NULL, contactnumber NUMBER(20) UNIQUE NOT NULL, type VARCHAR2(50) NOT NULL);	
Count of records in the table	**20**	

…

Name of the table	***schedule***			
Description	Stores the daily schedule of each staff member.			
Attribute	**Description**	**Type**	**Examples of values**	**Notes**
staffid	Id of the staff	number	1235	Not Null
scheduledate	Date of the schedule	date	09/21/2021	Not Null
starttime	Start time of the staff schedule	time	14:00	Not Null
endtime	End time of the staff schedule	time	21:00	Not Null
Functional Dependencies and Keys				
Functional dependencies	(staffid, scheduledate) → starttime, endtime			
Candidate keys	**(staffid, scheduledate)**			
Normalization				
1NF	**Yes**	All cells contain a unique value		

2NF	**Yes**	Table is in 1NF and no prime attribute is dependent on any proper subset of any candidate key of the table
3NF	**Yes**	All the non-key attributes depend only on a key
BCNF	**Yes**	Every functional dependency X → Y, X is the super key of the table
Physical Design		
Primary Key	(staffid, scheduledate)	
Foreign Keys	**staffid**	
SQL Code	CREATE TABLE schedule (staffid NUMBER(10) NOT NULL, scheduledate DATE NOT NULL, starttime TIMESTAMP WITH LOCAL TIME ZONE NOT NULL, endtime TIMESTAMP WITH LOCAL TIME ZONE NOT NULL, CONSTRAINT pk_schedule_id PRIMARY KEY (staffid,scheduledate), CONSTRAINT fk_staff_id FOREIGN KEY(staffid) REFERENCES staff(id));	
Count of records in the table	**23**	

…

Name of the table	***reservation***			
Description	Stores the reservation details of the customers.			
Attribute	**Description**	**Type**	**Examples of values**	**Notes**
id	Id of the reservation table	number	123456	Unique, Not null
customerid	Id of the customer made the reservation	number	345	Not null
datetime	time stamp of the reservation attempted	datetime	09-21-2019 14:30:00	Not null
status	Status of the reservation	varchar	success, waitlist, failed	Not null
waitlist	waitlist number in the reservation queue	number	2	Can be Null
partysize	Resevation made for number of people	number	4	Can be Null (If Status =Failed)
tableid	Id of the table reserved	number	1	Can be Null (If Status =Failed, waitlist)
Functional Dependencies and Keys				
Functional dependencies	id → customerid, datetime, status, waitlist, partysize, tableid			
Candidate keys	**id**			

Normalization		
1NF	**Yes**	All cells contain a unique value
2NF	**Yes**	Table is in 1NF and no prime attribute is dependent on any proper subset of any candidate key of the table
3NF	**Yes**	All the non-key attributes depend only on a key
BCNF	**Yes**	Every functional dependency X → Y, X is the super key of the table
Physical Design		
Primary Key	id	
Foreign Keys	**customerid, tableid**	
SQL Code	CREATE TABLE reservation (id NUMBER(10) PRIMARY KEY, customerid NUMBER(10) NOT NULL, datetime TIMESTAMP WITH LOCAL TIME ZONE NOT NULL, status VARCHAR2(50) NOT NULL, waitlist NUMBER(2), partysize NUMBER(2), tableid NUMBER(2), CONSTRAINT fk_reservecust_id FOREIGN KEY(customerid) REFERENCES customer(id), CONSTRAINT fk_reservetable_id FOREIGN KEY(tableid) REFERENCES tble(id));	
Count of records in the table	**50**	

…

Name of the table	***tble***			
Description	Table details in the restaurant.			
Attribute	**Description**	**Type**	**Examples of values**	**Notes**
id	Id of the table	number	1	Unique, Not null
status	Status of the table	varchar	Occupied, Empty	Not Null
capacity	Table capacity	number	4	Not Null
layout	Layout of the table	varchar	corner	Not Null
Functional Dependencies and Keys				
Functional dependencies	id → status, capacity, layout			
Candidate keys	**id**			
Normalization				
1NF	**Yes**	All cells contain a unique value		

2NF	**Yes**	Table is in 1NF and no prime attribute is dependent on any proper subset of any candidate key of the table
3NF	**Yes**	All the non-key attributes depend only on a key
BCNF	**Yes**	Every functional dependency X → Y, X is the super key of the table
Physical Design		
Primary Key	id	
Foreign Keys	-	
SQL Code	CREATE TABLE tble (id NUMBER(10) PRIMARY KEY, status VARCHAR2(50) NOT NULL, capacity NUMBER(2) NOT NULL, layout VARCHAR2(50) NOT NULL);	
Count of records in the table	**10**	

…

Name of the table	***orders***			
Description	Details of the order placed by the customer.			
Attribute	**Description**	**Type**	**Examples of values**	**Notes**
id	Id of the order	number	234	Unique, Not Null
customerid	Id of the customer placed.	number	456	Not Null
reservationid	Id of the reservation.	number	678	Not Null
datetime	Time stamp when the order is placed.	datetime	09-21-2019 14:30:00	Not Null
status	Status of the order	varchar	Served, Inprogress, cancelled	Not Null
Functional Dependencies and Keys				
Functional dependencies	id → customerid, reservationid, datetime, status			
Candidate keys	**id**			
Normalization				
1NF	**Yes**	All cells contain a unique value		
2NF	**Yes**	Table is in 1NF and no prime attribute is dependent on any proper subset of any candidate key of the table		
3NF	**Yes**	All the non-key attributes depend only on a key		
BCNF	**Yes**	Every functional dependency X → Y, X is the super key of the table		

Physical Design	
Primary Key	id
Foreign Keys	**customerid, reservationid**
SQL Code	CREATE TABLE orders (id NUMBER(10) PRIMARY KEY, customerid NUMBER(10) NOT NULL, reservationid NUMBER(10) NOT NULL, datetime TIMESTAMP WITH LOCAL TIME ZONE NOT NULL, status VARCHAR2(50) NOT NULL, CONSTRAINT fk_ordercust_id FOREIGN KEY(customerid) REFERENCES customer(id), CONSTRAINT fk_orderreserve_id FOREIGN KEY(reservationid) REFERENCES reservation(id));
Count of records in the table	**20**

...

Name of the table	***take***			
Description	Stores the staff id for each order.			
Attribute	**Description**	**Type**	**Examples of values**	**Notes**
staffid	Id of the staff	number	123	Not Null
orderid	Id of the order	number	456	Not Null
Functional Dependencies and Keys				
Functional dependencies				
Candidate keys	**(staffid, orderid)**			
Normalization				
1NF	**Yes**	All cells contain a unique value		
2NF	**Yes**	Table is in 1NF and no prime attribute is dependent on any proper subset of any candidate key of the table		
3NF	**Yes**	All the non-key attributes depend only on a key		
BCNF	**Yes**	Every functional dependency X → Y, X is the super key of the table		
Physical Design				
Primary Key	(staffid, orderid)			
Foreign Keys	**staffid, orderid**			

SQL Code	CREATE TABLE take (staffid NUMBER(10) NOT NULL, orderid NUMBER(10) NOT NULL, CONSTRAINT pk_take_id PRIMARY KEY (staffid,orderid), CONSTRAINT fk_takestaff_id FOREIGN KEY(staffid) REFERENCES staff(id), CONSTRAINT fk_takeorder_id FOREIGN KEY(orderid) REFERENCES orders(id));
Count of records in the table	**20**

…

Name of the table	***containedin***			
Description	Stores the menu id and quantity for each order			
Attribute	**Description**	**Type**	**Examples of values**	**Notes**
orderid	Id of the customer	number	123	Not Null
menuid	Id of the menu	number	156	Not Null
quantity	Quantity of the menu items ordered	number	3	Not Null
Functional Dependencies and Keys				
Functional dependencies	(orderid, menuid) → quantity			
Candidate keys	**(orderid, menuid)**			
Normalization				
1NF	**Yes**	All cells contain a unique value		
2NF	**Yes**	Table is in 1NF and no prime attribute is dependent on any proper subset of any candidate key of the table		
3NF	**Yes**	All the non-key attributes depend only on a key		
BCNF	**Yes**	Every functional dependency X → Y, X is the super key of the table		
Physical Design				
Primary Key	(menuid, orderid)			
Foreign Keys	**menuid, orderid**			

SQL Code	CREATE TABLE containedin (menuid NUMBER(10) NOT NULL, orderid NUMBER(10) NOT NULL, quantity NUMBER(10) NOT NULL, CONSTRAINT pk_containedin_id PRIMARY KEY (menuid,orderid), CONSTRAINT fk_containedinstaff_id FOREIGN KEY(menuid) REFERENCES menu(id), CONSTRAINT fk_containedinorder_id FOREIGN KEY(orderid) REFERENCES orders(id));
Count of records in the table	**43**

…

Name of the table	***menu***			
Description	Stores the menu details served in the restaurant.			
Attribute	**Description**	**Type**	**Examples of values**	**Notes**
id	Id of the menu	number	56897	Unique, Not null
itemname	Name of the item	varchar	Coffee	Not Null
categories	Category of the item	varchar	Beverages	Not Null
priceusd	Price of the item	float	9.99	Not Null
Functional Dependencies and Keys				
Functional dependencies	id → itemname, categories, priceusd			
Candidate keys	**id**			
Normalization				
1NF	**Yes**	All cells contain a unique value		
2NF	**Yes**	Table is in 1NF and no prime attribute is dependent on any proper subset of any candidate key of the table		
3NF	**Yes**	All the non-key attributes depend only on a key		
BCNF	**Yes**	Every functional dependency X → Y, X is the super key of the table		
Physical Design				
Primary Key	id			
Foreign Keys	**-**			

SQL Code	CREATE TABLE menu (id NUMBER(10) PRIMARY KEY, itenname VARCHAR2(50) NOT NULL, categories VARCHAR2(50) NOT NULL, priceusd FLOAT(5) NOT NULL);
Count of records in the table	**11**

…

Name of the table	***inventory***			
Description	Stores inventory of the items.			
Attribute	**Description**	**Type**	**Examples of values**	**Notes**
id	Id of the inventory	number	67595	Unique, Not Null
name	Name of the inventory	varchar	Coffee beans	Not Null
category	Category	varchar	Beans	Not Null
quantity	Quantity left in the inventory.	number	3	Not Null
Functional Dependencies and Keys				
Functional dependencies	id → name, category, quantity			
Candidate keys	**id**			
Normalization				
1NF	**Yes**	All cells contain a unique value		
2NF	**Yes**	Table is in 1NF and no prime attribute is dependent on any proper subset of any candidate key of the table		
3NF	**Yes**	All the non-key attributes depend only on a key		
BCNF	**Yes**	Every functional dependency X → Y, X is the super key of the table		
Physical Design				
Primary Key	id			
Foreign Keys	**-**			
SQL Code	CREATE TABLE inventory (id NUMBER(10) PRIMARY KEY, name VARCHAR2(50) NOT NULL, categories VARCHAR2(50) NOT NULL, quantity NUMBER(10) NOT NULL);			

Count of records in the table	**14**

...

Name of the table	***menuhasinventory***			
Description	Stores the inventory items for each menu item.			
Attribute	**Description**	**Type**	**Examples of values**	**Notes**
menuid	Id of the menu item	number	3453	Not Null
inventoryid	Id of the inventory item	number	5676	Not Null
Functional Dependencies and Keys				
Functional dependencies				
Candidate keys	**(menuid, inventoryid)**			
Normalization				
1NF	**Yes**	All cells contain a unique value		
2NF	**Yes**	Table is in 1NF and no prime attribute is dependent on any proper subset of any candidate key of the table		
3NF	**Yes**	All the non-key attributes depend only on a key		
BCNF	**Yes**	Every functional dependency X → Y, X is the super key of the table		
Physical Design				
Primary Key	(menuid, inventoryid)			
Foreign Keys	**menuid, inventoryid**			
SQL Code	CREATE TABLE menuhasinventory (menuid NUMBER(10) NOT NULL, inventoryid NUMBER(10) NOT NULL, CONSTRAINT pk_menuhasinventory_id PRIMARY KEY (menuid,inventoryid), CONSTRAINT fk_menuhasinventorymenu_id FOREIGN KEY(menuid) REFERENCES menu(id), CONSTRAINT fk_menuhasinventoryinventory_id FOREIGN KEY(inventoryid) REFERENCES inventory(id));			
Count of records in the table	**31**			

...

Name of the table	***payment***			
Description	Stores the payment details attempted by the customer.			
Attribute	**Description**	**Type**	**Examples of values**	**Notes**
id	Id of the payment	number	76853	Unique, Not Null
orderid	Id of the order	number	5646	Not Null
staffid	Id of the staff	number	76854	Not Null
datetime	Time stamp when payment is attempted	datetime	09-21-2019 14:30:00	Not Null
type	Mode of the payment by the customer	varchar	Cash, Credit Card	Not Null
status	Status of the payment	varchar	Success, Processing, Failed	Not Null
amountpaid	Amount paid by the customer	float	45	Not null
orderamount	Amount of the Order	float	40	Not Null
serviceprovider	Third Party Service partner of the payment	varchar	easypay	Can be null (Type=Cash)
Functional Dependencies and Keys				
Functional dependencies	id → orderid, staffid, datetime, type, status, amountpaid, orderamount, serviceprovider			
Candidate keys	**id**			
Normalization				
1NF	**Yes**	All cells contain a unique value		
2NF	**Yes**	Table is in 1NF and no prime attribute is dependent on any proper subset of any candidate key of the table		
3NF	**Yes**	All the non-key attributes depend only on a key		
BCNF	**Yes**	Every functional dependency X → Y, X is the super key of the table		
Physical Design				
Primary Key	id			
Foreign Keys	**orderid, staffid**			

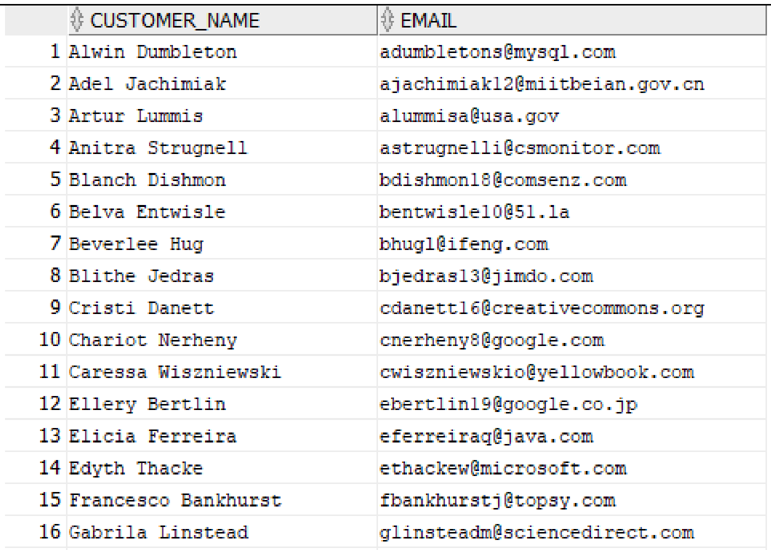
SQL Code	CREATE TABLE payment (id NUMBER(10) PRIMARY KEY, orderid NUMBER(10) NOT NULL, staffid NUMBER(10) NOT NULL, datetime TIMESTAMP WITH LOCAL TIME ZONE NOT NULL, type VARCHAR2(50) NOT NULL, status VARCHAR2(50) NOT NULL, amountpaid FLOAT(5) NOT NULL, orderamount FLOAT(5) NOT NULL, serviceprovider VARCHAR2(50), CONSTRAINT fk_paymentorder_id FOREIGN KEY(orderid) REFERENCES orders(id), CONSTRAINT fk_paymentstaff_id FOREIGN KEY(staffid) REFERENCES staff(id));
Count of records in the table	**9**

...

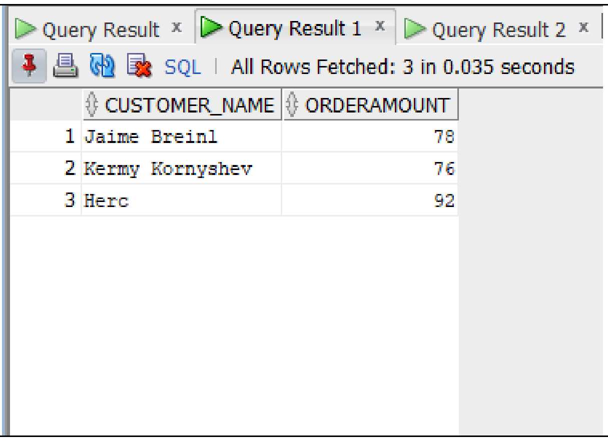
Name of the table	***carddetail***			
Description	Stores the card details for payment made by the non-cash type.			
Attribute	**Description**	**Type**	**Examples of values**	**Notes**
paymentid	Id of the payment	number	684567	Unique, Not Null
cardtype	Type of the Card	varchar	Credit, debit	Not Null
bankdetail	Name of the bank	varchar	PNC	Can be null
encryptedcardnumber	encrypted card number	varchar	xxxxxxxxxx7868	Not Null
Functional Dependencies and Keys				
Functional dependencies	paymentid → cardtype, bankdetail, encryptedcardnumber			
Candidate keys	**paymentid**			
Normalization				
1NF	**Yes**	All cells contain a unique value		
2NF	**Yes**	Table is in 1NF and no prime attribute is dependent on any proper subset of any candidate key of the table		
3NF	**Yes**	All the non-key attributes depend only on a key		
BCNF	**Yes**	Every functional dependency X → Y, X is the super key of the table		
Physical Design				
Primary Key	paymentid			
Foreign Keys	**paymentid**			

	SQL Code	CREATE TABLE carddetail (paymentid NUMBER(10) NOT NULL UNIQUE, cardtype VARCHAR2(50) NOT NULL, bankdetail VARCHAR2(50), encryptedcardnumber VARCHAR2(255) NOT NULL, CONSTRAINT fk_carddetailpayment_id FOREIGN KEY(paymentid) REFERENCES payment(id));
	Count of records in the table	**6**

MILESTONE 6: SQL QUERIES

<table>
<tr><th>Query 1</th><th></th></tr>
<tr><td><b>English version</b></td><td>DISPLAY ALL THE name of CUSTOMERS TO WHOM YOU CAN SEND TEXT EMAIL</td></tr>
<tr><td><b>SQL sentence</b></td><td><code>SELECT concat(concat(firstname, ' '), lastname) as Customer_Name, email FROM customer WHERE email IS NOT NULL ;</code></td></tr>
<tr><td><b>Example of returned rows (cropped screen caption)</b></td><td>
<table>
<tr><th></th><th>CUSTOMER_NAME</th><th>EMAIL</th></tr>
<tr><td>1</td><td>Alwin Dumbleton</td><td>adumbletons@mysql.com</td></tr>
<tr><td>2</td><td>Adel Jachimiak</td><td>ajachimiak12@miitbeian.gov.cn</td></tr>
<tr><td>3</td><td>Artur Lummis</td><td>alummisa@usa.gov</td></tr>
<tr><td>4</td><td>Anitra Strugnell</td><td>astrugnelli@csmonitor.com</td></tr>
<tr><td>5</td><td>Blanch Dishmon</td><td>bdishmon18@comsenz.com</td></tr>
<tr><td>6</td><td>Belva Entwisle</td><td>bentwisle10@51.la</td></tr>
<tr><td>7</td><td>Beverlee Hug</td><td>bhugl@ifeng.com</td></tr>
<tr><td>8</td><td>Blithe Jedras</td><td>bjedras13@jimdo.com</td></tr>
<tr><td>9</td><td>Cristi Danett</td><td>cdanett16@creativecommons.org</td></tr>
<tr><td>10</td><td>Chariot Nerheny</td><td>cnerheny8@google.com</td></tr>
<tr><td>11</td><td>Caressa Wiszniewski</td><td>cwiszniewskio@yellowbook.com</td></tr>
<tr><td>12</td><td>Ellery Bertlin</td><td>ebertlin19@google.co.jp</td></tr>
<tr><td>13</td><td>Elicia Ferreira</td><td>eferreiraq@java.com</td></tr>
<tr><td>14</td><td>Edyth Thacke</td><td>ethackew@microsoft.com</td></tr>
<tr><td>15</td><td>Francesco Bankhurst</td><td>fbankhurstj@topsy.com</td></tr>
<tr><td>16</td><td>Gabrila Linstead</td><td>glinsteadm@sciencedirect.com</td></tr>
</table></td></tr>
</table>

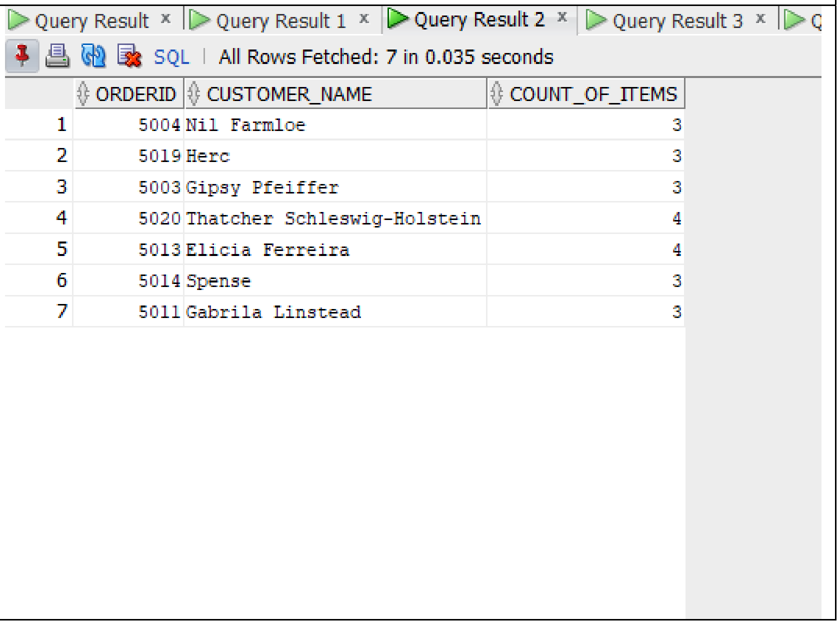
…

Query 2	
English version	Display all the name of customers whose order is more than $75
SQL sentence	SELECT concat(concat(c.firstname, ' '), c.lastname) as Customer_Name, p.orderamount FROM payment p INNER JOIN orders o ON p.orderid = o.id AND p.orderamount > 75 INNER JOIN customer c on o.customerid = c.id;
Example of returned rows (cropped screen caption)	 All Rows Fetched: 3 in 0.035 seconds CUSTOMER_NAME ORDERAMOUNT 1 Jaime Breinl 78 2 Kermy Kornyshev 76 3 Herc 92

…

Query 3	
English version	Display the orderid, name of customers, and count of items where the cusomter orders more than 2 items

SQL sentence	SELECT co.orderid, concat(concat(c.firstname,
	' '), c.lastname) as Customer_Name,
	COUNT(co.menuid) as Count_of_Items
	FROM orders o
	INNER JOIN containedin co ON o.id = co.orderid
	INNER JOIN customer c ON c.id = o.customerid
	GROUP BY co.orderid, c.firstname, c.lastname
	HAVING COUNT(co.menuid) > 2;

Example of returned rows (cropped screen caption)	

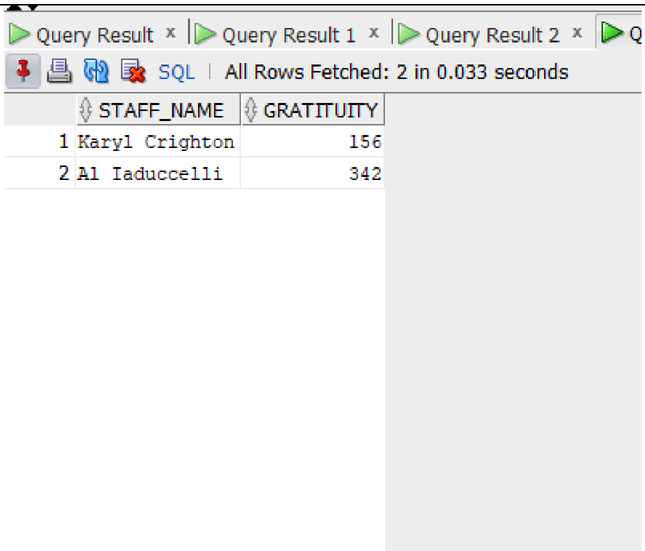
All Rows Fetched: 7 in 0.035 seconds

	ORDERID	CUSTOMER_NAME	COUNT_OF_ITEMS
1	5004	Nil Farmloe	3
2	5019	Herc	3
3	5003	Gipsy Pfeiffer	3
4	5020	Thatcher Schleswig-Holstein	4
5	5013	Elicia Ferreira	4
6	5014	Spense	3
7	5011	Gabrila Linstead	3

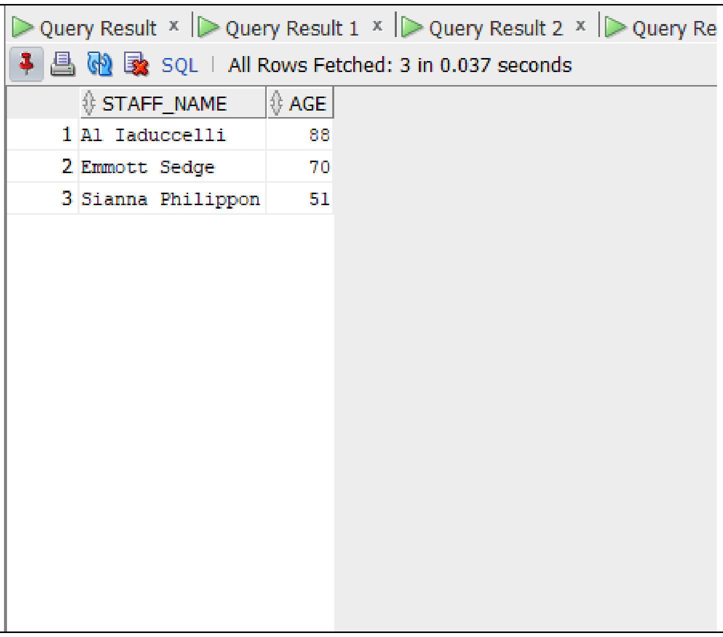
…

Query 4	
English version	Display the names of staff whose first name ends with 'L' and has received gratituity greater than $100

SQL sentence	`SELECT DISTINCT concat(concat(s.firstname, ' '), s.lastname) as Staff_Name,`
	`SUM(p.amountpaid - p.orderamount) as gratituity`
	`FROM payment p`
	`INNER JOIN staff s ON s.id = p.staffid`
	`WHERE s.firstname LIKE '%l' AND (p.amountpaid - p.orderamount) > 50`
	`GROUP BY s.firstname, s.lastname;`

Example of returned rows (cropped screen caption)	 Query Result ×
	Query Result 1 ×
	Query Result 2 ×
	Q
	SQL
	All Rows Fetched: 2 in 0.033 seconds
	`  STAFF_NAME       GRATITUITY`
	`1 Karyl Crighton          156`
	`2 Al Iaduccelli           342`

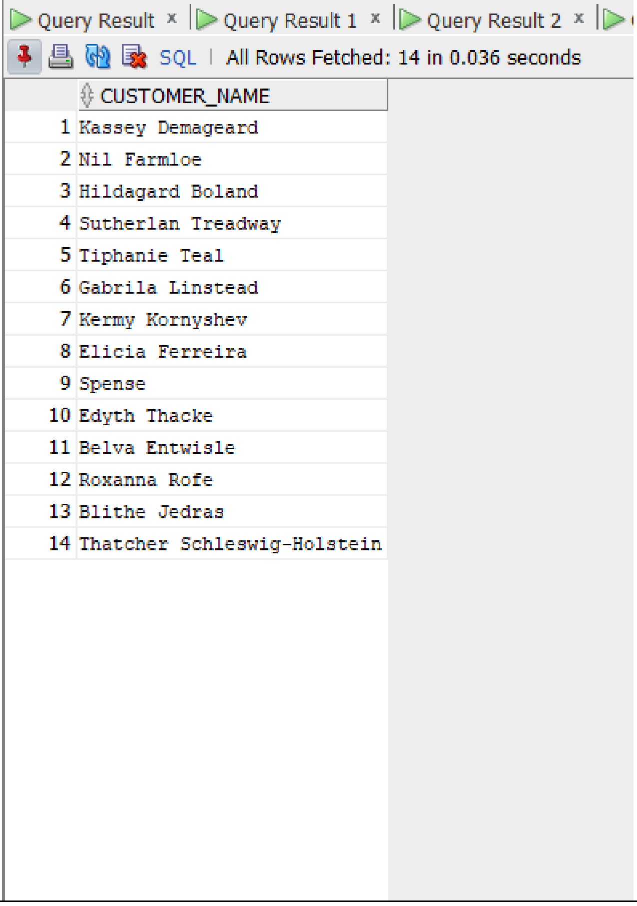
…

Query 5	
English version	Display the names and age of staff who took alcohol orders (age should be greater than 21)
SQL sentence	SELECT DISTINCT concat(concat(s.firstname, ' '), s.lastname) as Staff_Name, TRUNC((SYSDATE - TO_DATE(s.dateofbirth, 'DD-MON-YYYY'))/ 365) AS AGE FROM orders o LEFT JOIN take t on t.orderid = o.id LEFT JOIN staff s on s.id = t.staffid INNER JOIN containedin co on co.orderid = o.id INNER JOIN menu m on m.id = co.menuid AND m.categories = 'alcoholic beverages';
Example of returned rows (cropped screen caption)	 Query Result Query Result 1 Query Result 2 Query Re SQL All Rows Fetched: 3 in 0.037 seconds STAFF_NAME AGE 1 Al Iaduccelli 88 2 Emmott Sedge 70 3 Sianna Philippon 51

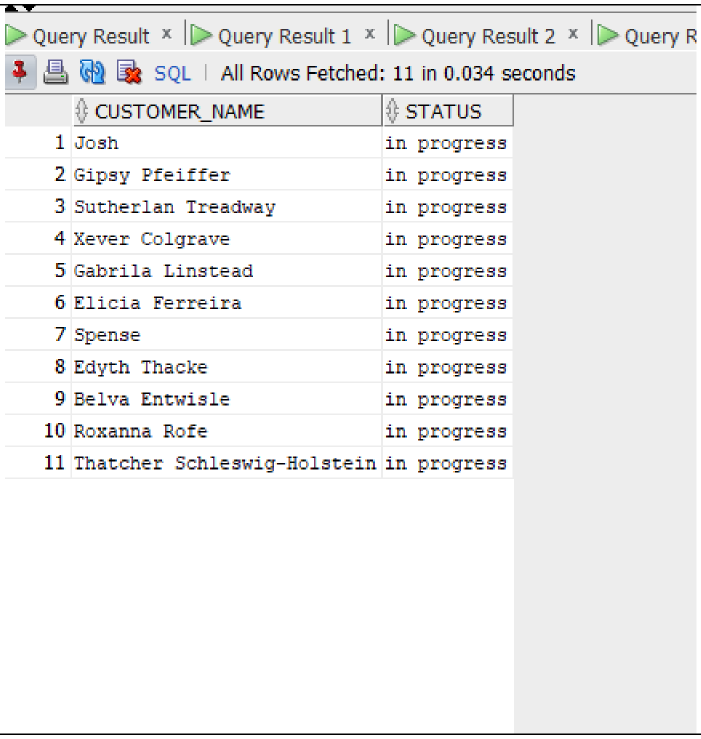
…

Query 6	
English version	Display the names of the customer who got center tables

SQL sentence	`SELECT concat(concat(c.firstname, ' '), c.lastname) as Customer_Name`
	`FROM reservation r`
	`INNER JOIN customer c ON c.id = r.customerid`
	`WHERE  r.status = 'success' AND r.tableid IN (SELECT  t.id FROM  tble t WHERE  t.layout = 'center');`

Example of returned rows (cropped screen caption)	 Query Result × Query Result 1 × Query Result 2 × SQL All Rows Fetched: 14 in 0.036 seconds CUSTOMER_NAME 1 Kassey Demageard 2 Nil Farmloe 3 Hildagard Boland 4 Sutherlan Treadway 5 Tiphanie Teal 6 Gabrila Linstead 7 Kermy Kornyshev 8 Elicia Ferreira 9 Spense 10 Edyth Thacke 11 Belva Entwisle 12 Roxanna Rofe 13 Blithe Jedras 14 Thatcher Schleswig-Holstein

…

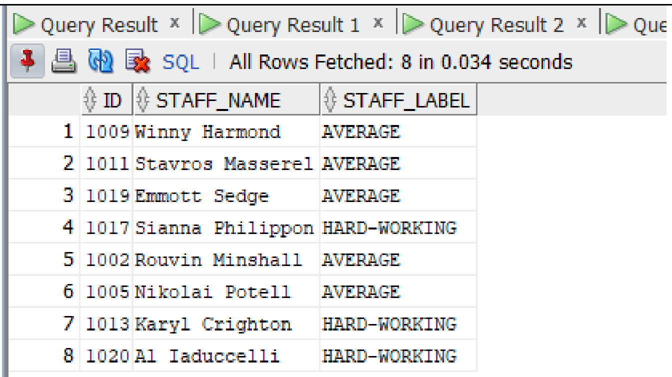
Query 7	
English version	Display the customer name whose order is in progess
SQL sentence	`SELECT (SELECT  concat(concat(c.firstname, ' '), c.lastname) FROM customer c WHERE customerid = id) as Customer_Name, o.status FROM orders o WHERE o.status = 'in progress';`
Example of returned rows (cropped screen caption)	 Query Result Query Result 1 Query Result 2 Query R SQL All Rows Fetched: 11 in 0.034 seconds

	CUSTOMER_NAME	STATUS
1	Josh	in progress
2	Gipsy Pfeiffer	in progress
3	Sutherlan Treadway	in progress
4	Xever Colgrave	in progress
5	Gabrila Linstead	in progress
6	Elicia Ferreira	in progress
7	Spense	in progress
8	Edyth Thacke	in progress
9	Belva Entwisle	in progress
10	Roxanna Rofe	in progress
11	Thatcher Schleswig-Holstein	in progress

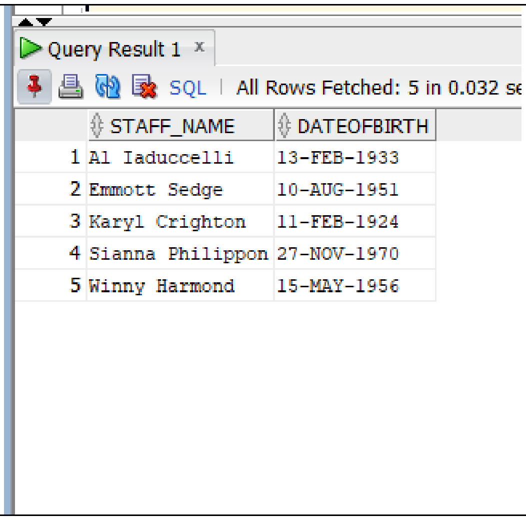
…

Query 8	
English version	Labelling name and id of staff based on the number of orders taken

SQL sentence	`SELECT s.id, concat(concat(s.firstname, ' '), s.lastname) as Staff_Name, CASE WHEN count(*) > 3 THEN 'HARD-WORKING' WHEN count(*) BETWEEN 1 AND 2 THEN 'AVERAGE' ELSE 'LAZY' END as Staff_Label`
	`FROM take t`
	`INNER JOIN staff s on s.id = t.staffid`
	`GROUP BY s.id,s.firstname, s.lastname;`

Example of returned rows (cropped screen caption)	 Query Result ×
	Query Result 1 ×
	Query Result 2 ×
	Que
	SQL
	All Rows Fetched: 8 in 0.034 seconds
	`  ID   STAFF_NAME        STAFF_LABEL`
	`1 1009 Winny Harmond     AVERAGE`
	`2 1011 Stavros Masserel  AVERAGE`
	`3 1019 Emmott Sedge      AVERAGE`
	`4 1017 Sianna Philippon  HARD-WORKING`
	`5 1002 Rouvin Minshall   AVERAGE`
	`6 1005 Nikolai Potell    AVERAGE`
	`7 1013 Karyl Crighton    HARD-WORKING`
	`8 1020 Al Iaduccelli     HARD-WORKING`

…

Query 9	
English version	Display the names of the staff who took have pending orders and verified the card payments done by visa cards
SQL sentence	SELECT DISTINCT concat(concat(s.firstname, ' '), s.lastname) as Staff_Name, s.dateofbirth FROM staff s, payment p, orders o, carddetail cd WHERE s.id=p.staffid AND o.id=p.orderid AND cd.cardtype = 'visa' UNION SELECT DISTINCT concat(concat(s.firstname, ' '), s.lastname) as Staff_Name, s.dateofbirth FROM staff s, payment p, orders o WHERE s.id=p.staffid AND o.id=p.orderid AND o.status = 'in progress';
Example of returned rows (cropped screen caption)	 Query Result 1 × SQL | All Rows Fetched: 5 in 0.032 se STAFF_NAME DATEOFBIRTH 1 Al Iaduccelli 13-FEB-1933 2 Emmott Sedge 10-AUG-1951 3 Karyl Crighton 11-FEB-1924 4 Sianna Philippon 27-NOV-1970 5 Winny Harmond 15-MAY-1956

…