

**DATABASE**

**SPECIFICATIONS**

Next Gen-Restaurant Application

*Rudraksh Mishra-* [*rjm7016@psu.edu*](mailto:rjm7016@psu.edu)

*Ambika Chundru-* [*ajc7832@psu.edu*](mailto:ajc7832@psu.edu)

**School of Graduate Professional Studies**

Information Science Department

IN SC 521 - Introduction to Database Concepts

October 2021 – December 2021

# Document Control

## Work carried out by:

|  |  |  |
| --- | --- | --- |
| **Name** | **Email Address** | **Other** |
| Rudraksh Mishra | rjm7016@psu.edu | Rudraksh.mishra04@gmail.com |
| Ambika Chundru | ajc7832@psu.edu | Ambika.chundru@gmail.com |
|  |  |  |

## Revision Sheet

|  |  |  |
| --- | --- | --- |
| **Release No.** | **Date** | **Revision Description** |
| **1** | **Nov 1St 2021** | Developed a core requirements table |
| 2 | Nov 8th 2021 | Developed a conceptual design and made updates to the first revision |
| 3 | Nov 15th 2021 | Developed a logical design and made updates to the second revision |
| 4 | Nov 21st 2021 | Normalized the tables and made updates to third revision |
| 5 | Dec 5th 2021 | Developed physical design and made the updates to fourth revision |
| 6 | Dec 9th 2021 | Developed SQL Queries and made the updates to the fifth revision |
|  |  |  |

**DATABASE SPECIFICATIONS**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Sr.No** | **Title** | **Page Number** |
| **1** | **Document Control** | **i** |
| **2** | **Milestone 1: Data Requirements** | **2** |
| **3** | **Milestone 2: Conceptual Design** | **4** |
| **4** | **Milestone 3: Logical Design** | **10** |
| **5** | **Milestone 4: Normalization** | **15** |
| **6** | **Milestone 5: Physical Design** | **16** |
| **7** | **Milestone 6: SQL queries** | **27** |

# Milestone 1: Data Requirements

## System Name or Title

Next Gen-Restaurant Application

## Core requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Requirement | Referenced page in SRS | Referenced Section in SRS | Referenced Paragraph in Section |
| 1 | The database should store information about customer. It needs to store information like customer name, customer id, phone number, DOB. | 3  6  11 | 1.2  2.3  3 | 1  2  3.5.3.3 |
| 2 | The party size, reservation time, reservation date, table number should also be stored when assigned. When there a no tables available the customer will be assigned a waiting list number. | 10  11  11  8 | 3  3  3  3 | 3.5.3.2  3.5.3.7  3.5.3.8  3.1.1 |
| 3 | The database should store unique id and user role to the users of the system. The user’s roles include: management members, servers, host/hostess, kitchen staff. Different type of users are given different authorizations. Each of the user’s information is stored using, username, password, name, DOB. | 6  8  10  13 | 2.3  3  3  5 | 2  3.1.2  3.5.3.1  1, 2 |
| 4 | The database should store the menu using item name, item id, cost per serving, item category. The item categories include entrée, side order, beverages, bar. | 19 | 8 | 8.1.1 |
| 5 | The order should be stored using order id, table number, customer id, item id, and item quantity. The status of the order should also be stored under order status. The statuses include order submitted, cancelled, order confirmation, preparing, ready for pick up and served. | 19  19 | 8  8 | 8.1.1  8.1.2 |
| 6 | The payment information should be stored using, ordered, payment id, amount paid, payment mode card details. We also need to store the id of the server assigned to the table, so that tip should be added to his payroll. | 9  9  13 | 3  3  5 | 3.5.1.5  3.5.1.9  5.3 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 7 | To edit the table map, we should store the properties of the table like, table id, table name, table capacity, status of the table, table shape, and table type | 10  32 | 3  8 | 3.5.2  8.1.14 |
| 8 | The database should store staff schedule. | 5 | 2 | 2 |
| 9 | Information about bar order should be stored. | 10 | 3 | 3.5.1.10 |
| 10 | Information about the ingredients should be stored. | 7 | 2.3 | table |

# Milestone 2: Conceptual Design

## Diagram

The below diagram will give a basic understanding on which entities are related. This will allow us to get a summarized view of the design:

Diagram

Description automatically generated

The below diagram will allow us to see how the entities are related. This gives us a detailed view of the systems design:

Diagram

Description automatically generated

## Below ER diagrams will give us the information about attributes in each of the entity.

**Table:**

Diagram

Description automatically generated

**Customer:**

A picture containing graphical user interface

Description automatically generated

**Staff:**

**A picture containing graphical user interface

Description automatically generated**

**Order:**

Diagram

Description automatically generated

## Transaction:

Diagram

Description automatically generated

**Menu:**

Diagram

Description automatically generated

## Inventory

## Diagram Description automatically generated

## Assumptions and Constraints

* One customer represents the entire group
* Menu depends on the inventory through the staff
* Order includes food orders and bar orders
* Staff with different roles have different authorizations
* Customers can book multiple tables if the table capacity is less than group size
* A bill can be split between multiple people in a group
* Menu has both food menu and bar menu
* One order is taken by multiple staff members, namely kitchen staff and server
* The stored password should be encrypted
* The item categories include entrée, side order, beverages, bar
* Only 2 staff members are authorized to edit the inventory, one personal deals with the food and the other deals with the bar. Category id in the inventory decides who deals with what.

# Milestone 3: Logical Design

## Entity Relationship Diagram

**Entity name**: Table

**Attributes**:

Id, type, capacity, shape, status

**Functional dependencies**:

Id → name, type, capacity, shape, status

|  |  |  |  |
| --- | --- | --- | --- |
| **Attributes not in FD** | **Attributes on the left** | **Attributes on both sides** | **Attributes on the right side** |
|  | id |  | name, type, capacity, shape, status |

**Attribute closures** (if any):

(Id)+= name, type, capacity, shape, status

**Unique keys**: the key for this table are

id

**Entity name**: Customer

**Attributes**:

id, password, name, phone no, address, dob

**Functional dependencies**:

id → password, name, phone no, address, dob

|  |  |  |  |
| --- | --- | --- | --- |
| **Attributes not in FD** | **Attributes on the left** | **Attributes on both sides** | **Attributes on the right side** |
|  | id |  | password, name, phone no, address, dob |

**Attribute closures** (if any):

(id)+= password, name, phone no, address, dob

**Unique keys**: the key for this table is

id

**Entity name**: staff

**Attributes**:

id, password, dob, name, role, phone number, address, email\_id,

**Functional dependencies**:

id → password, dob, name, role, phone number, address, email\_id,

|  |  |  |  |
| --- | --- | --- | --- |
| **Attributes not in FD** | **Attributes on the left** | **Attributes on both sides** | **Attributes on the right side** |
|  | id |  | password, dob, name, role, phone number, address, email\_id, |

**Attribute closures** (if any):

(Id) += password, dob, name, role, phone number, address,

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

id,

**Entity name**: Schedule

**Attributes**:

staff\_id, date, start\_time, out\_time

**Functional dependencies**:

staff\_id ,date→ start\_time, out\_time

|  |  |  |  |
| --- | --- | --- | --- |
| **Attributes not in FD** | **Attributes on the left** | **Attributes on both sides** | **Attributes on the right side** |
|  | Date  Staff\_id |  | start\_time, out\_time |

**Attribute closures** (if any):

(staff\_id, date)+=start\_time, out\_time

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

staff\_id, date (composite keys)

**Entity name**: order

**Attributes**:

id, item\_name, quantity, arrival\_time, type, status, reservation\_time,

**Functional dependencies**:

id → item\_name, quantity, arrival\_time, type, status, reservation\_time

|  |  |  |  |
| --- | --- | --- | --- |
| **Attributes not in FD** | **Attributes on the left** | **Attributes on both sides** | **Attributes on the right side** |
|  | id |  | item\_name, quantity, arrival\_time, type, status, reservation\_time |

**Attribute closures** (if any):

(Id)+= item\_name, quantity, arrival\_time, type, status, reservation\_time

**Unique keys**: the key for this table are

id

**Entity name**: transaction

**Attributes**:

Id, gratuities, total\_amount, pay\_cash, card\_type, card\_number

**Functional dependencies**:

id → gratuities, total\_amount, pay\_cash, card\_type, card\_number

|  |  |  |  |
| --- | --- | --- | --- |
| **Attributes not in FD** | **Attributes on the left** | **Attributes on both sides** | **Attributes on the right side** |
|  | id |  | gratuities, total\_amount, pay\_cash, card\_type, card\_number |

**Attribute closures** (if any):

(Id)+= gratuities, total\_amount, pay\_cash, card\_type, card\_number

**Unique keys**: the key for this table are

Id

**Entity name**: menu\_item

**Attributes**:

Id, name, category, price

**Functional dependencies**:

id → name, category, price

name → id, category, price

|  |  |  |  |
| --- | --- | --- | --- |
| **Attributes not in FD** | **Attributes on the left** | **Attributes on both sides** | **Attributes on the right side** |
|  | Id |  | category, price, name |

**Attribute closures** (if any):

(id )+= name, category, price

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

id

**Entity name**: inventory

**Attributes**:

Id, name, quatity,

**Functional dependencies**:

id → name, quatity,

|  |  |  |  |
| --- | --- | --- | --- |
| **Attributes not in FD** | **Attributes on the left** | **Attributes on both sides** | **Attributes on the right side** |
|  | Id |  | quatity, category\_id, name |

**Attribute closures** (if any):

(id )+= name, quatity, category\_id

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

Id

**Design**

**Diagram

Description automatically generated**

# Milestone 4: Normalization and

# Milestone 5: Physical Design

## Assumptions and Constraints

## 

## Naming Conventions

Discuss the naming standards and conventions that you have used for table creation.

## Tables

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ***Name of the table*** | ***Transactions*** | | | | |
|  | **Description** | An transaction is the way the customer pays bill for his/her order at restaurant | | | | |
|  | **Attribute** | **Description** | | **Type** | **Examples of values** | **Notes** |
|  | transaction\_id | Unique transaction number | | Varchar | T10000, T10001… |  |
|  | gratitude | Tip from customer to staff | | Integer | 10$ ,19$... |  |
|  | total\_amount | Total cost of order including taxes | | Integer | 20$,100$ |  |
|  | pay\_cash | Payment done by customer using cash | | Char | Yes, No |  |
|  | Card\_number | Number on the card which was used to pay the amount | | Integer | 64649279472,  52429346923.. |  |
|  | Card\_type | Type of the card used for the transactiopn | | Char | Credit, debit, … |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  | **Functional Dependencies and Keys** | | | | | |
|  | **Functional dependencies** | transaction\_id -> gratitude, total\_amount, pay\_cash, card\_number, card\_type | | | | |
|  | **Candidate keys** | transaction\_id | | | | |
|  | **Normalization** | | | | | |
|  | **1NF** | **Yes** | No arrays and repeated values nor any repeated column names | | | |
|  | **2NF** | **Yes** | No partial dependencies included | | | |
|  | **3NF** | **Yes** | No non-prime attribute is transitively dependent on the Candidate key | | | |
|  | **BCNF** | **Yes** | because all attributes depend only on the key (For trivial dependency in the relation X->Y, X is a super key of relation R) | | | |
|  | **Physical Design** | | | | | |
|  | **Primary Key** | **transaction\_id** | | | | |
|  | **Foreign Keys** | **order\_id, staff\_id, customer\_id** | | | | |
|  | **SQL Code** | CREATE TABLE transactions (  transaction\_id VARCHAR2(40) NOT NULL,  gratuities INTEGER,  total\_amount INTEGER NOT NULL,  pay\_cash VARCHAR(40),  card\_number INTEGER,  card\_type VARCHAR(40),  customer\_id VARCHAR2(40) NOT NULL,  order\_id VARCHAR2(40) NOT NULL,  staff\_id VARCHAR2(40) NOT NULL,  PRIMARY KEY (transaction\_id),  CONSTRAINT FK\_customertransaction FOREIGN KEY (customer\_id) REFERENCES customer(customer\_id),  CONSTRAINT FK\_orderstransaction FOREIGN KEY (order\_id) REFERENCES orders(order\_id),  CONSTRAINT FK\_stafftransaction FOREIGN KEY (staff\_id) REFERENCES staff(staff\_id)  ); | | | | |
|  | **Count of records in the table** | **30** | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ***Name of the table*** | ***Customer*** | | | | |
|  | **Description** | A customer is the user who orders at the restaurant. | | | | |
|  | **Attribute** | **Description** | | **Type** | **Examples of values** | **Notes** |
|  | customer\_id | Unique customer id | | Varchar | C10000, C10001… |  |
|  | name | Name of the customer | | Char | Jhon doe. Jane Mari |  |
|  | phone\_number | Phone number of the customer | | Integer | 7892445712, 9423885801 |  |
|  | password | A secret word or phrase that must be used to gain access to restaurant system | | Varchar | Hello@123,  Bie\_xyz,  11nov1996….. |  |
|  | address | Address of the customer | | Varchar | Apt no: r3, Xyz street, 190412 |  |
|  | dob | Date of birth of the customer | | Date | 11/01/1972,  03/24/1994 |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  | **Functional Dependencies and Keys** | | | | | |
|  | **Functional dependencies** | customer\_id -> name, phone\_number, password, address, dob | | | | |
|  | **Candidate keys** | **customer\_id** | | | | |
|  | **Normalization** | | | | | |
|  | **1NF** | **Yes** | No arrays and repeated values nor any repeated column names | | | |
|  | **2NF** | **Yes** | No partial dependencies included | | | |
|  | **3NF** | **Yes** | No non-prime attribute is transitively dependent on the Candidate key | | | |
|  | **BCNF** | **Yes** | because all attributes depend only on the key (For trivial dependency in the relation X->Y, X is a super key of relation R) | | | |
|  | **Physical Design** | | | | | |
|  | **Primary Key** | **customer\_id** | | | | |
|  | **Foreign Keys** | **-** | | | | |
|  | **SQL Code** | CREATE TABLE customer (  customer\_id VARCHAR2(20)NOT NULL,  cname VARCHAR2(40) NOT NULL,  phone\_number INTEGER,  cpassword VARCHAR2(40),  address VARCHAR2(40),  dob DATE,  PRIMARY KEY(customer\_id)  ); | | | | |
|  | **Count of records in the table** | **30** | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ***Name of the table*** | ***Orders*** | | | | |
|  | **Description** | Order is what the customer requests to be served | | | | |
|  | **Attribute** | **Description** | | **Type** | **Examples of values** | **Notes** |
|  | order\_id | Unique order number | | Varchar | O10001, O10002.. |  |
|  | status | Status of the order | | Char | confirmed, preparing, served |  |
|  | item\_name | Name of the item ordered | | Char | Pasta, pizza |  |
|  | quantity | Quantity of the item ordered | | Integer | 1, 2, 3… |  |
|  | type | Category of the order item | | Char | Food, Bar |  |
|  | arrival\_time | Time when the customer arrived at the restaurent | | Datetime | 2008-11-11 13:23:44 |  |
|  | reservation\_time | Reservation time of table of the cutomer | | Datetime | 2008-11-11 13:23:44 |  |
|  |  |  | |  |  |  |
|  | **Functional Dependencies and Keys** | | | | | |
|  | **Functional dependencies** | order\_id -> status, item\_name, quatity, type, arrival\_time, reservation\_time | | | | |
|  | **Candidate keys** | **order\_id** | | | | |
|  | **Normalization** | | | | | |
|  | **1NF** | **Yes** | No arrays and repeated values nor any repeated column names | | | |
|  | **2NF** | **Yes** | No partial dependencies included | | | |
|  | **3NF** | **Yes** | No non-prime attribute is transitively dependent on the Candidate key | | | |
|  | **BCNF** | **Yes** | because all attributes depend only on the key (For trivial dependency in the relation X->Y, X is a super key of relation R) | | | |
|  | **Physical Design** | | | | | |
|  | **Primary Key** | **order\_id** | | | | |
|  | **Foreign Keys** | **customer\_id** | | | | |
|  | **SQL Code** | CREATE TABLE orders (  order\_id VARCHAR2(40) PRIMARY KEY,  status VARCHAR2(40) ,  item\_name VARCHAR2(40),  quantity INTEGER,  arrival\_time TIMESTAMP,  reservation\_time TIMESTAMP NOT NULL,  item\_type VARCHAR2(40),  customer\_id VARCHAR2(40) NOT NULL,  CONSTRAINT FK\_customerorders FOREIGN KEY(customer\_id) REFERENCES customer(customer\_id)  ); | | | | |
|  | **Count of records in the table** | **30** | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ***Name of the table*** | ***Sit\_table*** | | | | |
|  | **Description** | An individual table at the restaurant | | | | |
|  | **Attribute** | **Description** | | **Type** | **Examples of values** | **Notes** |
|  | table\_id | Unique table number | | Varchar | T1, T2 … |  |
|  | table\_type | Type of the table | | Char | Large, Small.. |  |
|  | table\_capacity | Maximum number of people the table can hold | | Integer | 2,4,8,12.. |  |
|  | table\_shape | Shape of the table | | char | Square, Rectangle.. |  |
|  | table\_status | The status of the table | | Char | Reserved, Open, Occupied.. |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  | **Functional Dependencies and Keys** | | | | | |
|  | **Functional dependencies** | table\_id -> table\_type, table\_capacity, table\_shape, table\_status | | | | |
|  | **Candidate keys** | **id** | | | | |
|  | **Normalization** | | | | | |
|  | **1NF** | **Yes** | No arrays and repeated values nor any repeated column names | | | |
|  | **2NF** | **Yes** | No partial dependencies included | | | |
|  | **3NF** | **Yes** | No non-prime attribute is transitively dependent on the Candidate key | | | |
|  | **BCNF** | **Yes** | because all attributes depend only on the key (For trivial dependency in the relation X->Y, X is a super key of relation R) | | | |
|  | **Physical Design** | | | | | |
|  | **Primary Key** | **table\_id** | | | | |
|  | **Foreign Keys** | **order\_id** | | | | |
|  | **SQL Code** | CREATE TABLE sit\_table (  table\_id VARCHAR2(40) NOT NULL,  table\_type VARCHAR2(40) NOT NULL,  table\_capacity INTEGER NOT NULL,  table\_shape VARCHAR2(40) NOT NULL,  table\_status VARCHAR2(40) NOT NULL,  order\_id VARCHAR2(40),  PRIMARY KEY (table\_id),  CONSTRAINT FK\_orderstable FOREIGN KEY (order\_id) REFERENCES orders(order\_id)  ); | | | | |
|  | **Count of records in the table** | **30** | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ***Name of the table*** | ***Staff*** | | | | |
|  | **Description** | The authorized individuals who work at the restaurant | | | | |
|  | **Attribute** | **Description** | | **Type** | **Examples of values** | **Notes** |
|  | staff\_id | Unique staff number | | Varchar | S1001, S1002.. |  |
|  | sname | Name of the staff member | | Char | Jacob tyler, Jane Doe… |  |
|  | srole | The role of the customer in the restaurant | | Char | Chef, host, server, manger,… |  |
|  | address | Address of the staff | | Varchar | Apt no: r3, Xyz street, 190412 |  |
|  | phone\_number | Contact number of the staff | | Integer | 2453286293,  3467812649 |  |
|  | email\_id | Email id of the staff | | Varchar | Jane.d@gmail.com |  |
|  | dob | date of birth of the staff | | Date | 11/04/1997 |  |
|  | password | A secret word or phrase that must be used to gain access to restaurant system | | Varchar | Hello@123,  Bie\_xyz,  11nov1996….. |  |
|  | **Functional Dependencies and Keys** | | | | | |
|  | **Functional dependencies** | staff\_id -> sname, srole, address, phone\_number, email\_id, password  email\_id -> staff\_id, sname, srole, address, phone\_number, password | | | | |
|  | **Unique Key** | **email\_id** | | | | |
|  | **Candidate keys** | **staff\_id** | | | | |
|  | **Normalization** | | | | | |
|  | **1NF** | **Yes** | No arrays and repeated values nor any repeated column names | | | |
|  | **2NF** | **Yes** | partial dependencies included | | | |
|  | **3NF** | **Yes** | No non-prime attribute is transitively dependent on the Candidate key | | | |
|  | **BCNF** | **Yes** | because all attributes depend only on the key (For trivial dependency in the relation X->Y, X is a super key of relation R) | | | |
|  | **Physical Design** | | | | | |
|  | **Primary Key** | **staff\_id** | | | | |
|  | **Foreign Keys** | **-** | | | | |
|  | **SQL Code** | CREATE TABLE staff (  staff\_id VARCHAR2(20) NOT NULL,  sname VARCHAR2(40) NOT NULL,  srole VARCHAR2(40) NOT NULL,  email\_id VARCHAR2(40),  phone\_number INTEGER NOT NULL,  spassword VARCHAR2(40) NOT NULL,  address VARCHAR2(40)NOT NULL,  dob DATE NOT NULL,  PRIMARY KEY(staff\_id)  ); | | | | |
|  | **Count of records in the table** | **30** | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ***Name of the table*** | ***Menu\_item*** | | | | |
|  | **Description** | Items which are served in the restaurant upon request | | | | |
|  | **Attribute** | **Description** | | **Type** | **Examples of values** | **Notes** |
|  | menu\_item\_id | Unique menu item number | | Varchar | M10000, M10001 |  |
|  | menu\_item\_name | Name of the item which is available | | Char | Pasta, pizza |  |
|  | menu\_item\_category | The category of the item | | Char | Food, Bar |  |
|  | menu\_item\_price | Price if the item per serving | | Integer | 10$, 20$ |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  | **Functional Dependencies and Keys** | | | | | |
|  | **Functional dependencies** | menu\_item\_id -> menu\_item\_name, menu\_item\_category, menu\_item\_price | | | | |
|  | **Candidate keys** | **menu\_item\_id** | | | | |
|  | **Normalization** | | | | | |
|  | **1NF** | **Yes** | No arrays and repeated values nor any repeated column names | | | |
|  | **2NF** | **Yes** | No partial dependencies included | | | |
|  | **3NF** | **Yes** | No non-prime attribute is transitively dependent on the Candidate key | | | |
|  | **BCNF** | **Yes** | because all attributes depend only on the key (For trivial dependency in the relation X->Y, X is a super key of relation R) | | | |
|  | **Physical Design** | | | | | |
|  | **Primary Key** | **menu\_item\_id** | | | | |
|  | **Foreign Keys** | **-** | | | | |
|  | **SQL Code** | CREATE TABLE menu\_item (  menu\_item\_id VARCHAR2(40) NOT NULL,  menu\_item\_name VARCHAR2(40) NOT NULL,  menu\_item\_category VARCHAR2(40) NOT NULL,  menu\_item\_price INTEGER NOT NULL,  PRIMARY KEY(menu\_item\_id)  ); | | | | |
|  | **Count of records in the table** | **35** | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ***Name of the table*** | ***Inventory*** | | | | |
|  | **Description** | The raw items stored to make menu items | | | | |
|  | **Attribute** | **Description** | | **Type** | **Examples of values** | **Notes** |
|  | inventory\_item\_id | Unique menu item number | | Varchar | I10000,I10001… |  |
|  | iname | Name of the raw item | | Char | Tomato, Potato.. |  |
|  | quantity | The amount of the raw item available in the inventory | | varchar | 1kg , 2 kg… |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  | **Functional Dependencies and Keys** | | | | | |
|  | **Functional dependencies** | inventory\_item\_id -> inventory\_item\_name, inventory\_item\_quantity | | | | |
|  | **Candidate keys** | **inventory\_item\_id** | | | | |
|  | **Normalization** | | | | | |
|  | **1NF** | **Yes** | No arrays and repeated values nor any repeated column names | | | |
|  | **2NF** | **Yes** | No partial dependencies included | | | |
|  | **3NF** | **Yes** | No non-prime attribute is transitively dependent on the Candidate key | | | |
|  | **BCNF** | **Yes** | because all attributes depend only on the key (For trivial dependency in the relation X->Y, X is a super key of relation R) | | | |
|  | **Physical Design** | | | | | |
|  | **Primary Key** | **inventory\_item\_id** | | | | |
|  | **Foreign Keys** | **-** | | | | |
|  | **SQL Code** | CREATE TABLE inventory (  inventory\_item\_id VARCHAR2(40) NOT NULL,  iname VARCHAR2(20) NOT NULL,  quantity INTEGER,  PRIMARY KEY(inventory\_item\_id)  ); | | | | |
|  | **Count of records in the table** | 20 | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ***Name of the table*** | ***Schedule*** | | | | |
|  | **Description** | The date and timing when the staff works | | | | |
|  | **Attribute** | **Description** | | **Type** | **Examples of values** | **Notes** |
|  | staff\_id | Unique Staff id | | Varchar | S10000,S10001… |  |
|  | schedule\_date | Date associated with the staff working days | | Date | 11/02/2021 |  |
|  | start\_time | The intime when the staff should start working | | Time | 13:23:44 |  |
|  | out\_time | The out time when the staff can stop working | | Time | 13:23:44 |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  | **Functional Dependencies and Keys** | | | | | |
|  | **Functional dependencies** | staff\_id, date -> date, start\_time, out\_time | | | | |
|  | **Candidate keys** | **staff\_id, date** | | | | |
|  | **Normalization** | | | | | |
|  | **1NF** | **Yes** | No arrays and repeated values nor any repeated column names | | | |
|  | **2NF** | **Yes** | No partial dependencies included | | | |
|  | **3NF** | **Yes** | No non-prime attribute is transitively dependent on the Candidate key | | | |
|  | **BCNF** | **Yes** | because all attributes depend only on the key (For trivial dependency in the relation X->Y, X is a super key of relation R) | | | |
|  | **Physical Design** | | | | | |
|  | **Primary Key** | **staff\_id, date** | | | | |
|  | **Foreign Keys** | **staff\_id** | | | | |
|  | **SQL Code** | CREATE TABLE schedule (  staff\_id VARCHAR2(40) NOT NULL,  schedule\_date DATE NOT NULL,  start\_time TIMESTAMP NOT NULL,  out\_time TIMESTAMP NOT NULL,  PRIMARY KEY (staff\_id, schedule\_date),  CONSTRAINT FK\_staffschedule FOREIGN KEY (staff\_id) REFERENCES staff(staff\_id)  ); | | | | |
|  | **Count of records in the table** | **30** | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ***Name of the table*** | ***Manages*** | | | | |
|  | **Description** | Relationship between staff and inventory | | | | |
|  | **Attribute** | **Description** | | **Type** | **Examples of values** | **Notes** |
|  | staff\_id | Unique Staff id | | Varchar | S10000,S10001… |  |
|  | inventory\_item\_id | Unique inventory item id | | Varchar | I10000,I10001… |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  | **Functional Dependencies and Keys** | | | | | |
|  | **Functional dependencies** |  | | | | |
|  | **Candidate keys** | **staff\_id, inventory\_item\_id** | | | | |
|  | **Normalization** | | | | | |
|  | **1NF** |  |  | | | |
|  | **2NF** |  |  | | | |
|  | **3NF** |  |  | | | |
|  | **BCNF** |  |  | | | |
|  | **Physical Design** | | | | | |
|  | **Primary Key** | **staff\_id, inventory\_item\_id** | | | | |
|  | **Foreign Keys** | **staff\_id, inventory\_item\_id** | | | | |
|  | **SQL Code** | CREATE TABLE manages (  inventory\_item\_id VARCHAR2(40) NOT NULL,  staff\_id VARCHAR2(40) NOT NULL,  PRIMARY KEY(inventory\_item\_id, staff\_id),  CONSTRAINT FK\_inventorymanages FOREIGN KEY (inventory\_item\_id) REFERENCES inventory(inventory\_item\_id),  CONSTRAINT FK\_staffmanages FOREIGN KEY (staff\_id) REFERENCES staff(staff\_id)  ); | | | | |
|  | **Count of records in the table** | **30** | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ***Name of the table*** | ***Takes*** | | | | |
|  | **Description** | Relationship between order and staff | | | | |
|  | **Attribute** | **Description** | | **Type** | **Examples of values** | **Notes** |
|  | staff\_id | Unique Staff id | | Varchar | S10000,S10001… |  |
|  | order\_id | Unique order id | | Varchar | O10000,O10001… |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  | **Functional Dependencies and Keys** | | | | | |
|  | **Functional dependencies** |  | | | | |
|  | **Candidate keys** | **staff\_id, order\_id** | | | | |
|  | **Normalization** | | | | | |
|  | **1NF** |  |  | | | |
|  | **2NF** |  |  | | | |
|  | **3NF** |  |  | | | |
|  | **BCNF** |  |  | | | |
|  | **Physical Design** | | | | | |
|  | **Primary Key** | **staff\_id, order\_id** | | | | |
|  | **Foreign Keys** | **staff\_id, order\_id** | | | | |
|  | **SQL Code** | CREATE TABLE takes (  staff\_id VARCHAR2(40) NOT NULL,  order\_id VARCHAR2(40) NOT NULL,  PRIMARY KEY(staff\_id, order\_id),  CONSTRAINT FK\_stafftakes FOREIGN KEY (staff\_id) REFERENCES staff(staff\_id),  CONSTRAINT FK\_ordertakes FOREIGN KEY (order\_id) REFERENCES orders(order\_id)  ); | | | | |
|  | **Count of records in the table** | **30** | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ***Name of the table*** | ***Depends*** | | | | |
|  | **Description** | Relationship between order and menu items | | | | |
|  | **Attribute** | **Description** | | **Type** | **Examples of values** | **Notes** |
|  | order\_id | Unique order id | | Varchar | O10000,O10001… |  |
|  | menu\_item\_id | Unique menu item id | | Varchar | M10000,M10001… |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  |  |  | |  |  |  |
|  | **Functional Dependencies and Keys** | | | | | |
|  | **Functional dependencies** |  | | | | |
|  | **Candidate keys** | **order\_id, menu\_item\_id** | | | | |
|  | **Normalization** | | | | | |
|  | **1NF** |  |  | | | |
|  | **2NF** |  |  | | | |
|  | **3NF** |  |  | | | |
|  | **BCNF** |  |  | | | |
|  | **Physical Design** | | | | | |
|  | **Primary Key** | **order\_id, menu\_item\_id** | | | | |
|  | **Foreign Keys** | **order\_id, menu\_item\_id** | | | | |
|  | **SQL Code** | CREATE TABLE depends (  menu\_item\_id VARCHAR2(40) NOT NULL,  order\_id VARCHAR2(40) NOT NULL,  PRIMARY KEY(menu\_item\_id, order\_id),  CONSTRAINT FK\_menuitemdepends FOREIGN KEY (menu\_item\_id) REFERENCES menu\_item(menu\_item\_id),  CONSTRAINT FK\_ordersdepends FOREIGN KEY (order\_id) REFERENCES orders(order\_id)  ); | | | | |
|  | **Count of records in the table** | **30** | | | | |

# Milestone 6: SQL queries and

|  |  |
| --- | --- |
| **Query 1** |  |
| **English version** | Display all the staff who came to work on 1st Nov 2021(Staff schedule support) |
| **Source for the query need in the SRS document** | SRS document, Page 5, 2.2, 6th point |
| **SQL sentence** | SELECT staff.sname  FROM schedule  INNER JOIN staff ON schedule.staff\_id=staff.staff\_id WHERE schedule.schedule\_date = '01-Nov-21' ; |
| **Example of returned rows (cropped screen caption)** | Text  Description automatically generated |

|  |  |
| --- | --- |
| **Query 2** |  |
| **English version** | Display all the names and phone numbers of customers to whom text message be sent regarding their reservation |
| **Source for the query need in the SRS document** | SRS document, Page 33, Test case ID: 8.1.15 |
| **SQL sentence** | SELECT cname, phone\_number FROM customer WHERE phone\_number IS NOT NULL ; |
| **Example of returned rows (cropped screen caption)** | Table  Description automatically generated |

|  |  |
| --- | --- |
| **Query 3** |  |
| **English version** | Display the most ordered items and which category do they belong to |
| **Source for the query need in the SRS document** |  |
| **SQL sentence** | SELECT item\_name, COUNT(\*), item\_type  FROM orders  GROUP BY item\_name, item\_type  HAVING COUNT(\*) > 2 |
| **Example of returned rows (cropped screen caption)** | Table  Description automatically generated with medium confidence |

|  |  |
| --- | --- |
| **Query 4** |  |
| **English version** | Display the names of the customers who gave gratuities used a card to do the payment |
| **Source for the query need in the SRS document** |  |
| **SQL sentence** | SELECT DISTINCT C.cname  FROM transactions T  RIGHT JOIN customer C ON T.customer\_id=C.customer\_id WHERE gratuities IS NOT NULL AND pay\_cash IS NULL; |
| **Example of returned rows (cropped screen caption)** | Graphical user interface, text, application  Description automatically generated |

|  |  |
| --- | --- |
| **Query 5** |  |
| **English version** | Display the names and dob of the servers who took Bar orders (To check staff age is above 18)and verified the card payments done by jbc cards |
| **Source for the query need in the SRS document** |  |
| **SQL sentence** | SELECT DISTINCT S.sname, S.dob  FROM staff S, transactions T, orders O  WHERE S.staff\_id=T.staff\_id AND O.order\_id=T.order\_id AND T.card\_type = 'jbc'  UNION  SELECT DISTINCT S.sname, S.dob  FROM staff S, transactions T, orders O  WHERE S.staff\_id=T.staff\_id AND O.order\_id=T.order\_id AND O.item\_type = 'Bar'; |
| **Example of returned rows (cropped screen caption)** | Table  Description automatically generated |

|  |  |
| --- | --- |
| **Query 6** |  |
| **English version** | Display the roles and the names of the staff members who can edit the inventory |
| **Source for the query need in the SRS document** |  |
| **SQL sentence** | SELECT DISTINCT S.srole, S.sname  FROM staff S, inventory I, manages M  WHERE S.staff\_id=M.staff\_id AND I.inventory\_item\_id=M.inventory\_item\_id; |
| **Example of returned rows (cropped screen caption)** | Table  Description automatically generated |

|  |  |
| --- | --- |
| **Query 7** |  |
| **English version** | Display the names of the customers who canceled they orders |
| **Source for the query need in the SRS document** |  |
| **SQL sentence** | SELECT DISTINCT C.cname  FROM customer C  WHERE C.customer\_id IN ( SELECT O.customer\_id  FROM orders O  WHERE O.status='Cancelled'); |
| **Example of returned rows (cropped screen caption)** | Text, application  Description automatically generated |

|  |  |
| --- | --- |
| **Query 8** |  |
| **English version** | Display the id of the table whose table status is occupied, and the order status is served. So that we can process their billing |
| **Source for the query need in the SRS document** |  |
| **SQL sentence** | SELECT T.table\_id, O.status, t.table\_status  FROM sit\_table T  INNER JOIN orders O ON T.order\_id=o.order\_id WHERE t.table\_status LIKE 'O%' AND O.status LIKE 'S%' ; |
| **Example of returned rows (cropped screen caption)** | A picture containing text  Description automatically generated |

|  |  |
| --- | --- |
| **Query 9** |  |
| **English version** | Group and display the table based on their type and shape |
| **Source for the query need in the SRS document** |  |
| **SQL sentence** | SELECT table\_type, table\_shape, COUNT(\*)  FROM sit\_table  GROUP BY table\_type, table\_shape; |
| **Example of returned rows (cropped screen caption)** | Table  Description automatically generated |