0. Pre-Illumination

In this project report we will follow the requirement of Phase IV directly. In Section 1 we gave problem description copied from Web site; in Section 2 we answered 3 questions listed in the project and justified our solution; in Section 3 we exhibited EER diagram with all assumptions; in Section 4 we showed our relational schema after normalization; in Section 5 we gave all requested SQL statements for both views and queries; and in Section 6 we gave dependency diagram induced from relational schemas. Finally, a short summary is given at the end of this report.

1. Problem Description

See the Phase 0.

2. Three Questions

2.1 1. Is the ability to model super-class / subclass relationships likely to be important in such environment? Why or why not?

Ans: Super-class, subclass is very important in our database design. For example, in the hotel, we have lots of different types, such as dining, concierge, housekeeping, etc. Without the super-class/ subclass, we would have to use weak entity type to represent the different types, which could lead to redundance. Now, we use subclasses of employee types and thus, our database achieves BCNF.

2.2

1. There are two rates of membership: Silver Prime & Golden Prime

2. There is another type of employee, shuttle driver. A shuttle driver has an experience year.

3. The hotel has an account in the bank. The accountant either has the authorization to manage this account or not. The account records the profit and cost daily.

4. There is a type of employee as procurement. They purchase goods needed for the hotel.

5. There is a type of employee as advertisement. A member in this apartment has an experience year.

2.3 Justify using a Relational DBMS like Oracle for this project.

See figure 1.

Figure 1: Oracle database implementation for the hotel database design.

3. EER diagram with all assumptions

Assumptions

1. The reason why we do not use a total constraint for employee is that there are other job titles not listed.
2. We assumed that an employee cannot have multiple job titles.
3. We assumed that individuals cannot overlap with organization. If an individual client wants to register as an organization, we will give him/her a new Customer_ID.
4. We assumed that the there are other type of work serve as an event staff, so we do not use a total constraint.
5. We assumed that our customer can hold a membership card but never check in.

4. Relational Schema in Third Normal Form

4.1 Regular Entity

1. Employee:

EmployeeID	Age	Salary_rate	Name	State	City	Street_name	Street_number	Zipcode

2. Dinning:

EmployeeID	Shift	Type

EmployeeID is a FK referencing to the Employee.EmployeeID.

3. Concierge:

EmployeeID	Concierge_experience_year

EmployeeID is a FK referencing to the Employee.EmployeeID.

4. Housekeeping:

EmployeeID	Housekeeping_experience_year

EmployeeID is a FK referencing to the Employee.EmployeeID.

5. Receptionist:

EmployeeID	Language

EmployeeID is a FK referencing to the Employee.EmployeeID.

6. Tech_support:

EmployeeID	Tech_support_licence

EmployeeID is a FK referencing to the Employee.EmployeeID.

7. Management:

EmployeeID	Title

EmployeeID is a FK referencing to the Employee.EmployeeID.

8. Catering:

EmployeeID

EmployeeID is a FK referencing to the Employee.EmployeeID.

9. Accountant:

EmployeeID	Accountant_licence

EmployeeID is a FK referencing to the Employee.EmployeeID.

---

10. Event:

Event_ID	Event_date	Event_time	Event_manager_ID	Event_deposit

Event_manager_ID is a FK referencing to the Management.EmployeeID.

11. Client:

Customer_ID

12. Organization:

Customer_ID

Customer_ID is a FK referencing to the Client. Customer_ID.

13. Individual:

Customer_ID	Birthdate	Individual_name	Individual_sex

Customer_ID is a FK referencing to the Client. Customer_ID.

---

14. Membership:

Membership_number

15. Room:

Room_number	Bed_type	Room_type	Per_night_price

2.2: Weak Entity (We do not have this☹)

2.3: Mapping 1:1 relationship

Employee:

EmployeeID	Age	Salary_rate	Name	State	City	Street_name	Street_number	Zipcode	Event_staff_ID

Event_staff_ID is a FK referencing to the Employee. EmployeeID.

2.4 Mapping 1:N relationship (We do not have this☹)

2.5 Mapping M:N relationship

16. Clear:

Employee_ID	Room_number	Date	Time

Employee_ID is a FK referencing to the Housekeeping.EmployeeID.

Room_number is a FK referencing to the Room. Room_number

17. Check_out_bill:

Room_number	Customer_ID	Check_out_bill_amount	Check_out_bill_issued_date	Check_in_date	Check_out_date

Room_number is a FK referencing to the Room. Room_number.

Customer_ID is a FK referencing to the Individual. Customer_ID.

18. Has:

Customer_ID	Membership_number

Customer_ID is a FK referencing to the Individual. Customer_ID.

Membership_number is a FK referencing to the Membership. Membership_number.

19. Holds:

Event_ID	Customer_ID

Event_ID is a FK referencing to the Event.Event_ID.

Customer_ID is a FK referencing to the Organization.Customer_ID.

20. Serve:

Event_ID	Employee_ID	On_call_speaker	On-call_number

Event_ID is a FK referencing to the Event.Event_ID.

Employee_ID is a FK referencing to the Employee.Employee_ID.

2.6: Mapping of Multivalued Attributes

21. Bill_event_payment:

(Implemented after 2.7)

Customer_ID	Event_bill_ID	Date	Time	Amount_of_event_payment	Date_of_payment

Customer_ID is a FK referencing to the Organization.Customer_ID.

22. Direct_bill_account:

Customer_ID	Direct_bill_account

Customer_ID is a FK referencing to the Organization.Customer_ID.

23. Phone:

Customer_ID	Phone

Customer_ID is a FK referencing to the Individual.Customer_ID.

24. Check_out_payment_ID:

Customer_ID	Room_number	COP_time	COP_date	COP_amount_of_payment

Customer_ID is a FK referencing to the Individual.Customer_ID.

Room_number is a FK referencing to the Room.Room_number.

2.7: Mapping n-ary relationship

25. Check_in:

Customer_ID	Employee_ID	Room_number	Check_in_date	Key_type	Check_in_time	Lounge	Length_of_stay

Customer_ID is a FK referencing to the Client.Customer_ID.

Employee_ID is a FK referencing to the Receptionist.Employee_ID.

Room_number is a FK referencing to the Room.Room_number.

26. Bill_for_event:

Customer_ID	Event_bill_ID	Event_ID	Employee_ID	Bill_event_date_issued	Bill_event_amount

Customer_ID is a FK referencing to the Organization.Customer_ID.

Event_ID is a FK referencing to the Event.Event_ID.

Employee_ID is a FK referencing to the Accountant.Employee_ID.

4.1 Explanation for format design

Assumption:

- Data format for all Employee ID is an integer with length of 9.
- Age is an integer without any further restriction.
- Salary_rate is a number within [0, 9999999.99].
- Event_deposit is a number within [0, 99999.99]. (Max deposit is roughly one hundred thousand dollars.)
- Room number is a 4 digits number.
- Membership number is a 11 digits number.
- Direct bill account is a 20 char-long string.
- Length of stay is an integer greater than 0.

4.2 Format for Every Relation

1.

Relation Name	Attributes	Data Type
	EmployeeID	Numeric(9,0)
	Age	Integer
	Salary_rate	Numeric(9,2)
	Name	String<=20

Employee	State	String<=2
	City	String<=20
	Street_name	String<=20
	Street_number	String<=20
	Zipcode	String<=10
	Event_staff_ID	Numeric(9,0)

2.

Relation Name	Attributes	Data Type
Dinning	EmployeeID	Numeric(9,0)
	Shift	String<=10
	Type	String<=20

3.

Relation Name	Attributes	Data Type
Concierge	EmployeeID	Numeric(9,0)
	Concierge_experience_year	Interger

4.

Relation Name	Attributes	Data Type
Housekeeping	EmployeeID	Numeric(9,0)
	Housekeeping_experience_year	Interger

5.

Relation Name	Attributes	Data Type
Receptionist	EmployeeID	Numeric(9,0)
	Language	String<=10

6.

Relation Name	Attributes	Data Type
Tech_support	EmployeeID	Numeric(9,0)
	Tech_support_licence	String<=100

7.

Relation Name	Attributes	Data Type
Management	EmployeeID	Numeric(9,0)
	Title	String<=20

8.

Relation Name	Attributes	Data Type
Catering	EmployeeID	Numeric(9,0)

9.

Relation Name	Attributes	Data Type

Accountant	EmployeeID	Numeric(9,0)
	Accountant_licence	String<=20

10.

Relation Name	Attributes	Data Type
Event	Event_ID	Numeric(4,0)
	Event_date	MM/DD/YYYY, STRING = 10 CHARS
	Event_time	hh-mm-ss, string = 8 chars
	Event_manager_ID	Numeric(9,0)
	Event_deposit	Numeric(7,2)

11.

Relation Name	Attributes	Data Type
Client	CustomerID	Numeric(6,0)

12.

Relation Name	Attributes	Data Type
Organization	CustomerID	Numeric(6,0)

13.

Relation Name	Attributes	Data Type
Individual	CustomerID	Numeric(6,0)
	Birthdate	MM/DD/YYYY, STRING = 10 CHARS
	Individual_name	String<=20
	Individual_sex	String<=20

14.

Relation Name	Attributes	Data Type
Membership	Membership_number	Numeric(11,0)

15.

Relation Name	Attributes	Data Type
Room	Room_number	Numeric(4,0)
	Bed_type	String<=20
	Room_type	String<=20
	Per_night_price	Integer

16.

Relation Name	Attributes	Data Type
Clear	EmployeeID	Numeric(9,0)
	Room_number	Numeric(4,0)
	cDate	MM/DD/YYYY, STRING = 10 CHARS

	cTime	hh-mm-ss, string = 8 chars

17.

Relation Name	Attributes	Data Type
Check_out_bill	Room_number	Numeric(4,0)
	CustomerID	Numeric(6,0)
	Check_out_bill_amount	Integer
	Check_out_bill_issued_date	MM/DD/YYYY, STRING = 10 CHARS
	Check_in_date	MM/DD/YYYY, STRING = 10 CHARS
	Check_out_date	MM/DD/YYYY, STRING = 10 CHARS

18.

Relation Name	Attributes	Data Type
Has	CustomerID	Numeric(6,0)
	Membership_number	Numeric(11,0)

19.

Relation Name	Attributes	Data Type
Holds	CustomerID	Numeric(6,0)
	Event_ID	Numeric(4,0)

20.

Relation Name	Attributes	Data Type
Serve	Event_ID	Numeric(4,0)
	EmployeeID	Numeric(9,0)
	On_call_speaker	String<=20
	On-call_number	Numeric(4,0)

21.

Relation Name	Attributes	Data Type
Bill_event_payment	CustomerID	Numeric(6,0)
	Event_bill_ID	Numeric(6,0)
	pDate	MM/DD/YYYY, STRING = 10 CHARS
	pTime	hh-mm-ss, string = 8 chars
	Amount_of_event_payment	Numeric(8,2)
	Date_of_payment	MM/DD/YYYY, STRING = 10 CHARS

22.

Relation Name	Attributes	Data Type
	CustomerID	Numeric(6,0)

Direct_bill_account	Direct_bill_account	String=20

23.

Relation Name	Attributes	Data Type
Phone	CustomerID	Numeric(6,0)
	Phone	Numeric(10,0)

24.

Relation Name	Attributes	Data Type
check_out_payment_id	CustomerID	Numeric(6,0)
	Room_number	Numeric(4,0)
	COP_time	hh-mm-ss, string = 8 chars
	COP_date	MM/DD/YYYY, STRING = 10 CHARS
	COP_amount_of_payment	Numeric(8,2)

25.

Relation Name	Attributes	Data Type
Check_in	CustomerID	Numeric(6,0)
	EmployeeID	Numeric(9,0)
	Room_number	Numeric(4,0)
	Check_in_date	MM/DD/YYYY, STRING = 10 CHARS
	Key_type	String<=20
	Check_in_time	hh-mm-ss, string = 8 chars
	Lounge	Boolean
	Length_of_stay	Integer>0

26.

Relation Name	Attributes	Data Type
Bill_for_event	CustomerID	Numeric(6,0)
	Event_bill_ID	Numeric(6,0)
	Event_ID	Numeric(4,0)
	EmployeeID	Numeric(9,0)
	Bill_event_date_issued	MM/DD/YYYY, STRING = 10 CHARS
	Bill_event_amount	Numeric(8,2)

5 All Requested SQL Statements

5.1 Creation of Database with SQL Statements

5.1.1 Table Creation

--1

```
create table employee
(employeeid      numeric(9,0)      primary key,
age        number(3)         not null,
salary_rate        numeric(9,2),
name    varchar2(20),
state     varchar2(2),
city       varchar2(20),
street_name      varchar2(20),
street_number  varchar2(20),
zipcode varchar2(10),
event_staff_id   numeric(9,0) null
)

alter table employee add constraint fk_employee
            foreign key (event_staff_id)        references employee (employeeid);

--2
create table dinning
(shift     varchar2(10)      null,
type      varchar2(20)      null,
employeeid numeric(9,0)    primary key
)

alter table dinning add constraint fk_dinning
            foreign key (employeeid)            references employee (employeeid);

--3
create table concierge
(
```

```
concierge_experience_year number(3)   null,
employeeid numeric(9,0)    primary key
)

alter table concierge add constraint fk_concierge
        foreign key (employeeid)        references employee (employeeid);

--4
create table housekeeping
(
housekeeping_experience_year number(3)        null,
employeeid numeric(9,0)    primary key
)

alter table housekeeping add constraint fk_housekeeping
        foreign key (employeeid)        references employee (employeeid);

--5
create table receptionist
(
language varchar2(10)   null,
employeeid numeric(9,0)    primary key
)

alter table receptionist add constraint fk_receptionist
        foreign key (employeeid)        references employee (employeeid);

--6
create table tech_support
```

```
(
tech_support_licence varchar2(10)        null,
employeeid numeric(9,0)    primary key
)

alter table tech_support add constraint fk_tech
        foreign key (employeeid)        references employee (employeeid);

--7
create table management
(
title      varchar2(20)      null,
employeeid numeric(9,0)    primary key
)

alter table management add constraint fk_management
        foreign key (employeeid)        references employee (employeeid);


--8
create table catering
(
employeeid numeric(9,0)    primary key
)

alter table catering add constraint fk_catering
        foreign key (employeeid)        references employee (employeeid);
```

```
--9
create table accountant
(
accountant_licence        varchar2(20)      null,
employeeid numeric(9,0)    primary key
)

alter table accountant add constraint fk_accountant
            foreign key (employeeid)            references employee (employeeid);

--10
create table event
(
event_id            numeric(4,0) primary key,
event_date        date,
event_time        timestamp(6) with time zone,
event_manager_id         numeric(9,0) not null,
event_deposit                numeric(7,2)
)

alter table event add constraint fk_event
            foreign key (event_manager_ID)            references employee (employeeid);

-- 11. Client
CREATE TABLE client
(
   customerid NUMERIC(6,0) PRIMARY KEY
);
```

```
-- 12. Organization
CREATE TABLE organization
(
   customerid NUMERIC(6,0) PRIMARY KEY
);

ALTER TABLE organization add CONSTRAINT fk_organization
            FOREIGN KEY (customerid)        REFERENCES client (customerid);

-- 13. Individual
CREATE TABLE individual
(
   customerid NUMERIC(6,0) PRIMARY KEY,
   birthdate DATE NULL,
   individual_name VARCHAR2(20) NULL,
   individual_sex VARCHAR2(20) NULL
);

ALTER TABLE individual add CONSTRAINT fk_individual
            FOREIGN KEY (customerid)        REFERENCES client (customerid);

-- 14. Membersip
CREATE TABLE membership
(
   membership_number NUMERIC(11,0) PRIMARY KEY
);

-- 15. Room
CREATE TABLE room
```

```
(
   room_number NUMERIC(4,0) PRIMARY KEY,
   bed_type VARCHAR2(20) NULL,
   room_type VARCHAR2(20) NULL,
   per_night_price INTEGER NULL
);

-- 16. Clear
CREATE TABLE clear
(
   employeeid NUMERIC(9,0) NOT NULL,
   room_number NUMERIC(4,0) NOT NULL,
   cdate DATE NULL,
   ctime TIMESTAMP(6) WITH TIME ZONE,
   PRIMARY KEY (employeeid, room_number)
);

ALTER TABLE clear add CONSTRAINT fk_clear1
         FOREIGN KEY (employeeid)        REFERENCES employee (employeeid);

ALTER TABLE clear add CONSTRAINT fk_clear2
         FOREIGN KEY (room_number)   REFERENCES room (room_number);

-- 17. check_out_bill
CREATE TABLE check_out_bill
(
   room_number NUMERIC(6,0) NOT NULL,
   customerid NUMERIC(6,0) NOT NULL,
   check_out_bill_amount INTEGER NOT NULL,
```

```
   check_out_bill_issed_date DATE NOT NULL,
   check_in_date DATE NOT NULL,
   check_out_date DATE NOT NULL,
   PRIMARY KEY (room_number, customerid)
);

-- 18. has
CREATE TABLE has
(
   customerid NUMERIC(6,0) NOT NULL,
   membership_number NUMERIC(11,0) NOT NULL,
   PRIMARY KEY (customerid, membership_number)
);

ALTER TABLE has add CONSTRAINT fk_has1
   FOREIGN KEY (customerid)    REFERENCES client (customerid);

ALTER TABLE has add CONSTRAINT fk_has2
   FOREIGN KEY (membership_number) REFERENCES membership (membership_number);

--19
create table holds
(
customerid        number(6,0) not null,
event_id          number(4,0) not null,
primary key (customerid, event_id)
)

alter table holds add constraint fk_holds
```

```
        foreign key (customerid)        references Client(customerid);

alter table holds add constraint fk_holds2
        foreign key (event_id)    references event (event_id);

--20
create table serve
(
event_id        number(4,0) not null,
employeeid      number(9,0) not null,
oncall_speaker          char(20) null,
oncall_number           number(4,0),
primary key (event_id, employeeid)
)

alter table serve add constraint fk_serve1
        foreign key (employeeid)        references employee (employeeid);

alter table serve add constraint fk_serve2
        foreign key (event_id)    references event (event_id);

//////////////////////////////////////////
--21
create table bill_event_payment
(
customerid      numeric(6,0),
event_bill_id   numeric(6,0),
pdate   date,
ptime TIMESTAMP(6) WITH TIME ZONE,
```

```
date_of_payment          date,
amount_of_event_payment numeric(8,2),
primary key (customerid, event_bill_id)
)

alter table bill_event_payment add constraint fk_bill_event_payment
            foreign key (customerid)            references organization (customerid);

-- 22. direct_bill_account
CREATE TABLE direct_bill_account
(
   customerid NUMERIC(6,0) NOT NULL,
   direct_bill_account VARCHAR2(20) NOT NULL,
   PRIMARY KEY (customerid, direct_bill_account)
);

ALTER TABLE direct_bill_account add CONSTRAINT fk_direct_bill_account
   FOREIGN KEY (customerid)    REFERENCES client (customerid);

-- 23. phone
CREATE TABLE phone
(
   customerid NUMERIC(6,0) PRIMARY KEY,
   phone NUMERIC(10,0) NULL
)

ALTER TABLE phone add CONSTRAINT fk_phone
   FOREIGN KEY (customerid)    REFERENCES client (customerid);
```

```
-- 24. check_out_payment_id
CREATE TABLE check_out_payment_id
(
   customerid NUMERIC(6,0) NOT NULL,
   room_number NUMERIC(4,0) NOT NULL,
   cop_time TIMESTAMP(6) WITH TIME ZONE,
   cop_date DATE NOT NULL,
   cop_amount_of_payment NUMERIC(8,2),
   PRIMARY KEY (customerid, room_number)
);

ALTER TABLE check_out_payment_id add CONSTRAINT fk_check_out_payment_id1
   FOREIGN KEY (customerid)    REFERENCES client (customerid);

ALTER TABLE check_out_payment_id add CONSTRAINT fk_check_out_payment_id2
   FOREIGN KEY (room_number)   REFERENCES room (room_number);

--25
create table check_in
(
customerid        numeric(6,0) not null,
employeeid        numeric(9,0) not null,
room_number   numeric(4,0) not null,
check_in_date   date not null,
key_type           char(20),
check_in_time   TIMESTAMP WITH LOCAL TIME ZONE not null,
length_of_stay  numeric(9,0),
lounge  numeric(1,0)  default '0',
primary key (customerid, employeeid, room_number, check_in_date, check_in_time)
```

```
)

alter table check_in add constraint fk_check_in1
        foreign key (employeeid)        references receptionist(employeeid);

alter table check_in add constraint fk_check_in2
        foreign key (customerid)        references client(customerid);

alter table check_in add constraint fk_check_in3
        foreign key (room_number)       references room(room_number);

--26
create table bill_for_event
(
customerid      number(6,0) not null,
event_bill_id   number(6,0),
event_id        number(4,0) not null,
employeeid      number(9,0) not null,
bill_event_date_issued          date,
bill_event_amount               number(8,2),
primary key (customerid, event_bill_id)
)

alter table bill_for_event add constraint fk_bill_for_event1
        foreign key (employeeid)        references accountant(employeeid);

alter table bill_for_event add constraint fk_bill_for_event2
        foreign key (customerid)        references organization (customerid);
```

alter table bill_for_event add constraint fk_bill_for_event3

foreign key (event_id) references event (event_id);

5.1.2 A Database State

--1. employee

	EMPLOYEEID	AGE	SALARY_RATE	NAME	STATE	CITY	STREET_NAME	STREET_NUMBER	ZIPCODE	EVENT_STAFF_ID
1	111111111	15	1500	li	il	chicago	35th	3548	60609	111111111
2	111111112	25	2000	tom	tx	richardson	renner	800	75080	111111112
3	111111113	35	1500	bill	ca	chicago	47th	456	60615	111111113
4	111111114	45	3000	zong	ny	new york	jackson	658	75657	111111114
5	111111115	55	5000	ali	ny	new york	martin	435	75754	111111115
6	111111116	65	2000	john	ca	l.a.	21th	8908	75865	111111116
7	111111117	75	3000	bob	tx	richardson	55th	4444	75567	111111117
8	111111118	85	2000	jim	tx	richardson	90th	9767	75524	111111118

--2. dinning

	SHIFT	TYPE	EMPLOYEEID
1	morning	lounge	111111111

--3. concierge

	EMPLOYEEID
1	111111117

--4. housekeeping

	HOUSEKEEPING_EXPERIENCE_YEAR	EMPLOYEEID
1	3	111111113

--5. receptionist

	LANGUAGE	EMPLOYEEID
1	english	111111114

--6. tech_support

	TECH_SUPPORT_LICENCE	EMPLOYEEID
1	wasd	111111115

--7. management

	TITLE	EMPLOYEEID
1	housekeeping manager	111111116

--8. catering

	EMPLOYEEID
1	111111117

--9. accountant

	ACCOUNTANT_LICENCE	EMPLOYEEID
1	asdf	111111118

--10. event

	EVENT_ID	EVENT_DATE	EVENT_TIME	EVENT_MANAGER_ID	EVENT_DEPOSIT
1	8888	17-NOV-90	01-NOV-17 11.15.30.000000000 AM AMERICA/CHICAGO	111111111	55.55
2	9999	18-NOV-91	01-NOV-17 12.16.31.000000000 PM AMERICA/CHICAGO	111111112	56.66

-- 11. client

	CUSTOMERID
1	111111
2	222222
3	333333
4	444444

-- 12. Organization

	CUSTOMERID
1	333333
2	444444

-- 13. Individual

	CUSTOMERID	BIRTHDATE	INDIVIDUAL_NAME	INDIVIDUAL_SEX
1	111111	01-NOV-95	danly	male
2	222222	03-DEC-96	nancy	female

-- 14. Membersip

	MEMBERSHIP_NUMBER
1	12345678901
2	12345678902

-- 15. Room

	ROOM_NUMBER	BED_TYPE	ROOM_TYPE	PER_NIGHT_PRICE
1	1234	king	standard	50
2	1235	queen	premium	50
3	1101	king	suite	35
4	1102	double	standard	55

-- 16. Clear

	EMPLOYEEID	ROOM_NUMBER	CDATE	CTIME
1	111111113	1234	17-NOV-90	01-NOV-17 11.15.30.000000000 AM AMERICA/CHICAGO
2	111111113	1235	17-NOV-90	01-NOV-17 11.15.30.000000000 AM AMERICA/CHICAGO

-- 17. check_out_bill

	ROOM_NUMBER	CUSTOMERID	CHECK_OUT_BILL_AMOUNT	CHECK_OUT_BILL_ISSED_DATE	CHECK_IN_DATE	CHECK_OUT_DATE
1	1234	111111	50	17-NOV-90	16-NOV-90	17-NOV-90

-- 18. has

	CUSTOMERID	MEMBERSHIP_NUMBER
1	111111	12345678901
2	222222	12345678902

--19. holds

	CUSTOMERID	EVENT_ID
1	333333	9999
2	444444	8888

--20. serve

	EVENT_ID	EMPLOYEEID	ONCALL_SPEAKER	ONCALL_NUMBER
1	9999	111111111	wsedfthyu	1357
2	8888	111111111	xcsdfhgj	1649

--21. bill_event_payment

	CUSTOMERID	EVENT_BILL_ID	PDATE	PTIME	DATE_OF_PAYMENT	AMOUNT_OF_EVENT_PAYMENT
1	333333	987654	11-NOV-11	01-NOV-17 12.13.14.000000000 PM AMERICA/CHICAGO	01-DEC-11	123456
2	444444	123456	12-NOV-13	01-NOV-17 11.12.14.000000000 AM AMERICA/CHICAGO	01-DEC-13	234567

-- 22. direct_bill_account

	CUSTOMERID	DIRECT_BILL_ACCOUNT
1	111111	qerwet
2	222222	fjoeaigj

-- 23. phone

	CUSTOMERID	PHONE
1	111111	800000000
2	222222	233333333

-- 24. check_out_payment_id

	CUSTOMERID	ROOM_NUMBER	COP_TIME	COP_DATE	COP_AMOUNT_OF_PAYMENT
1	111111	1234	11-DEC-12 11.50.00.000000000 PM AMERICA/CHICAGO	11-OCT-12	10000
2	222222	1235	21-DEC-13 11.50.00.000000000 PM AMERICA/CHICAGO	22-OCT-12	19999

--25. check_in

	CUSTOMERID	EMPLOYEEID	ROOM_NUMBER	CHECK_IN_DATE	KEY_TYPE	CHECK_IN_TIME	LENGTH_OF_STAY	LOUNGE
1	111111	111111114	1234	10-OCT-99	asdff	01-NOV-17 11.11.11.000000000 AM	10	0
2	222222	111111114	1235	10-OCT-00	rfwegbr	01-NOV-17 11.11.11.000000000 AM	9	0
3	111111	111111114	1235	17-DEC-00	fwwfe	01-NOV-17 11.11.11.000000000 AM	8	0

--26. bill_for_event

	CUSTOMERID	EVENT_BILL_ID	EVENT_ID	EMPLOYEEID	BILL_EVENT_DATE_ISSUED	BILL_EVENT_AMOUNT
1	333333	222222	9999	111111118	11-NOV-99	100000
2	444444	111111	8888	111111118	12-DEC-00	233333

5.2 Creation of Views (Answer for Question d/Phase III)

1.

```
CREATE VIEW available_room (room_number) AS

SELECT r.room_number

FROM room r, check_in ci
```

WHERE r.room_number != ci.room_number

	ROOM_NUMBER
1	1234
2	1101
3	1102
4	1235

2.

```
CREATE VIEW popular_event_manager (event_staff_id) AS
select event_manager_id
from event
where TO_CHAR(event_date, 'MMDD') >= TO_CHAR(SYSDATE - 30, 'MMDD')
group by event_manager_id
having count(event_id) > 10
```

Columns | Data | Grants | Dependencie

Sort.. | F

EVENT_STAFF_ID

3.

```
CREATE VIEW frequent_customer (customerID) AS
SELECT i.customerID
FROM individual i, check_in ci
WHERE  trunc(ci.Check_in_date, 'YEAR') = trunc(sysdate,'YEAR') and i.customerID = ci.customerID
GROUP BY i.customerID
having COUNT(ci.Check_in_date) >= 10
```

4.

```
CREATE VIEW popular_room (room_number) AS
SELECT r.room_number
FROM check_in ci, room r
WHERE r.room_number = ci.room_number and trunc(ci.check_in_date, 'YEAR') > trunc(sysdate,'YEAR')
GROUP BY r.room_number
having count (ci.check_in_date) > 30
```

5.3 Creation of SQL Queries (Answer for Question e/Phase III)

```
--1
select Count(employeeid) as nofEmployees
from dinning
where type = 'lounge' or type = 'bar'
```

	NOFEMPLOYEES
1	1

```
-- 2
SELECT AVG(Salary_rate)
```

```
FROM employee
WHERE employee.employeeid IN
(SELECT employeeid
FROM receptionist)
```

	AVG(SALARY_RATE)
1	3000

```
--3
select *
from individual i
where i.CUSTOMERID in (
select customerid
from (

select c.customerid, Sum(c.COP_AMOUNT_OF_PAYMENT)
from check_out_payment_id c
where TRUNC(c.cop_date, 'YEAR') = TRUNC(SYSDATE, 'YEAR')
group by c.customerid
having Sum(c.COP_AMOUNT_OF_PAYMENT) > 1000
))
```

Result: empty

```
-- 4
SELECT Check_out_bill_amount
FROM Check_out_bill
ORDER BY Check_in_date ASC
```

	CHECK_OUT_BILL_AMOUNT
1	50

```
--5
select *
from individual i
where i.CUSTOMERID in (

select customerid
from (
select c.customerid, Sum(c.length_of_stay)
from check_in c
where TRUNC(c.check_in_date, 'YEAR') = TRUNC(SYSDATE, 'YEAR')
group by c.customerid
having Sum(c.length_of_stay) >= 15
)
);
```

Result: empty

```
-- 6
select avg(k.AGE_TODAY)
from
(
select d.customerid, TRUNC((SYSDATE - TO_DATE(d.bd, 'YYYY-MM-DD'))/ 365.25) AS AGE_TODAY
from(
select birthdate as bd, i.customerid
from individual i
where i.customerid in
(
select customerid
```

```
from check_in c
where c.employeeid in
(
select r.employeeid
from receptionist r
where r.language = 'Spanish'
))
) d
) k
```

	AVG(K.AGE_TODAY)
1	(null)

```
--7
select *
from organization o
where o.customerid in
(
select h.customerid
from holds h
where h.customerid in
(
select b.customerid
from Bill_event_payment b
group by b.CUSTOMERID
having Sum(b.amount_of_event_payment) > 2000
)
group by h.customerid
having Count(Event_id) >= 2
);
```

Result: empty

```
-- 8
select *
from(
select b.bill_event_amount
from bill_for_event b
where b.event_id in
(
select event_id
from POPULAR_EVENT_MANAGER, event
)
order by b.bill_event_amount desc
)
where rownum =  1
```

Result: empty

```
--9
select *
from event e
where e.event_id in (
select event_id
from(
select b.event_id, b.customerid, max(bill_event_amount)
from bill_for_event b
group by b.event_id, b.customerid
))
```

	EVENT_ID	EVENT_DATE	EVENT_TIME	EVENT_MANAGER_ID	EVENT_DEPOSIT
1	8888	17-NOV-90	01-NOV-17 11.15.30.000000000 AM AMERICA/CHICAGO	111111111	55.55
2	9999	18-NOV-91	01-NOV-17 12.16.31.000000000 PM AMERICA/CHICAGO	111111112	56.66

```
-- 10
select *
from(
select c.check_in_date
from
(
select *
from
(
select e.room_number, Count(e.room_number) as shax
from check_in e
group by e.room_number
order by shax desc
)
where rownum = 1
) d, check_in c

where d.room_number = c.room_number
order by c.check_in_date desc
)
where rownum = 1
```

	CHECK_IN_DATE
1	17-DEC-00

6. Dependency Diagram

Employee:

EmployeeID	Age	Salary_rate	Name	State	City	Street_name	Street_number	Zipcode	Event_staff_ID

EmployeeID → Age, Salary_rate, Name, State, City, Street_name, Street_number, Zipcode, Event_staff_ID

Dinning:

EmployeeID	Shift	Type

EmployeeID → Shift, Type

Concierge:

EmployeeID	Concierge_experience_year

EmployeeID → Concierge_experience_year

Housekeeping:

EmployeeID	Housekeeping_experience_year

EmployeeID → Housekeeping_experience_year

Receptionist:

EmployeeID	Language

EmployeeID → Language

Tech_support:

EmployeeID	Tech_support_licence

EmployeeID → Tech_support_licence

Management:

EmployeeID	Title

EmployeeID → Title

Catering:

EmployeeID

Accountant:

EmployeeID	Accountant_licence

EmployeeID → Accountant_licence

Event:

Event_ID	Event_date	Event_time	Event_manager_ID	Event_deposit

Event_ID → Event_date, Event_time, Event_manager_ID, Event_deposit

Client:

Customer_ID

Organization:

Customer_ID

Individual:

Customer_ID	Birthdate	Individual_name	Individual_sex

Customer_ID → Birthdate, Individual_name, Individual_sex

Membership:

Membership_number

Room:

Room_number	Bed_type	Room_type	Per_night_price

Room_number → Bed_type, Room_type, Per_night_price

Clear:

Employee_ID	Room_number	Date	Time

Employee_ID, Room_number → Date, Time

Check_out_bill:

Room_number	Customer_ID	Check_out_bill_amount	Check_out_bill_issued_date	Check_in_date	Check_out_date

Room_number, Customer_ID → Check_out_bill_amount, Check_out_bill_issued_date, Check_in_date, Check_out_date

Has:

Customer_ID	Membership_number

Holds:

Event_ID	Customer_ID

Serve:

Event_ID	Employee_ID	On_call_speaker	On-call_number

Event_ID, Employee_ID → On_call_speaker, On-call_number

Bill_event_payment:

Customer_ID	Event_bill_ID	Date	Time	Amount_of_event_payment	Date_of_payment

Customer_ID, Event_bill_ID → Date, Time, Amount_of_event_payment, Date_of_payment

Direct_bill_account:

Customer_ID	Direct_bill_account

Phone:

Customer_ID	Phone

Check_out_payment_ID:

Customer_ID	Room_number	COP_time	COP_date	COP_amount_of_payment

Room_number, Customer_ID → Check_out_bill_amount, Check_out_bill_issued_date, Check_in_date, Check_out_date

Check_in:

Customer_ID	Employee_ID	Room_number	Check_in_date	Key_type	Check_in_time	Lounge	Length_of_stay

Customer_ID, Room_number, COP_time, COP_date → COP_amount_of_payment

Bill_for_event:

Customer_ID	Event_bill_ID	Event_ID	Employee_ID	Bill_event_date_issued	Bill_event_amount

Customer_ID, Event_bill_ID → Event_ID, Employee_ID, Bill_event_date_issued, Bill_event_amount

7. Conclusion

In this final report we summarized all the necessary descriptions and solutions for the hotel database, including process and result of EER diagrams, relational schemas in third normal form, SQL statements to create database, create view and solve corresponding queries, as well as dependency diagram. We also implement the whole database in Oracle and using a database state to test every query. In section 2 we also explained why we use superclass/subclass relationship to build relational schema, why we choose a Relational DBMS to implement our database, and the additional five business rules shown from implementation.