

2203588DONG Bo Database Systems

by Bo Dong

Submission date: 06-Jan-2023 08:00PM (UTC+0000)

Submission ID: 193847577

File name: 42017_Bo_Dong_2203588DONG_Bo_Database_Systems_474126_1988037499.docx (46.48K)

Word count: 1461

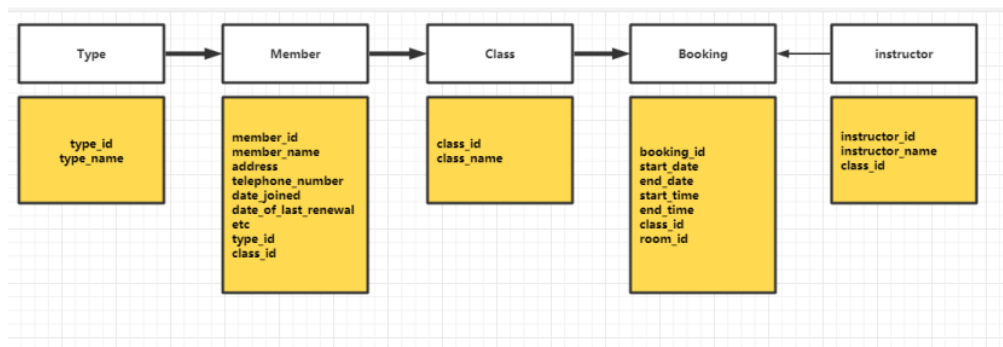
Character count: 8686

Abertawe Leisure Centre

Member Database Report And Implementation

A. Structure

1. Entity Relationship Diagram:



2. Create a PostgreSQL database

```
SQL
create database alc;
```

3. Tables and desc the solutions are providing to ALC

Type(type_id, type_name) Primary key : type_id

Member(member_id, member_name, address, telephone_number, date_joined, date_of_last_renewal, Etc, type_id, class_id) Primary key : member_id Foreign key : type_id

Class(class_id, class_name) Primary key : class_id

Booking(booking_id, start_date, end_date, start_time, end_time, class_id, room_id) Primary key : booking_id Foreign key : room_id

instructor(instructor_id, instructor_name, class_id) Primary key : instructor_id Foreign key : class_id

4. Create data dictionary

Table	Column	type	Length	Is null	Remark
Type	type_id	varchar	20	NO	PK
	type_name	varchar	20	NO	
Member	member_id	varchar	20	NO	PK
	member_name	varchar	20	YES	
	address	varchar	50	YES	
	telephone_number	varchar	20	NO	
	date_joined	date		NO	
	date_of_last_renewal	date		NO	
	etc	varchar	20	NO	
	type_id	varchar	20	NO	FK
	class_id	varchar	20	NO	
Class	class_id	varchar	20	NO	PK
	class_name	varchar	20	NO	
Booking	booking_id	varchar	20	NO	PK
	start_date	date		NO	
	end_date	date		NO	
	start_time	varchar	20	NO	
	end_time	varchar	20	NO	
	class_id	varchar	20	NO	
	room_id	varchar	20	NO	FK

instructor	instructor_id	varchar	20	NO	PK
	instructor_name	varchar	20	NO	
	class_id	varchar	20	NO	FK

5. Create table structure

```
SQL
-- Type
create table type
(
type_id varchar(20) primary key,
type_name varchar(20)
);
-- Member
create table Member
(
member_id varchar(20) primary key,
member_name varchar(20) ,
address varchar(50),
telephone_number varchar(20),
date_joined date,
date_of_last_renewal date,
etc varchar(20),
type_id varchar(20),
class_id varchar(20)
);
-- Class
create table Class
(
class_id varchar(20) primary key,
class_name varchar(20)
);
-- Booking
create table Booking
(
booking_id varchar(20) primary key,
start_date date,
end_date date,
```

```

start_time varchar(20),
end_time varchar(20),
class_id varchar(20),
room_id varchar(20)
);

-- instructor
create table instructor
(
teacher_id varchar(20) primary key,
teacher_name date,
class_id date
);

```

B. Written Elements

1. About Sql file

- a. SQL code used to generate each table structure;

```

SQL
-- Type
create table type
(
type_id varchar(20) primary key,
type_name varchar(20)
);
-- Member
create table Member
(
member_id varchar(20) primary key,
member_name varchar(20) ,
address varchar(50),
telephone_number varchar(20),
date_joined date,
date_of_last_renewal date,
etc varchar(20),
type_id varchar(20),
class_id varchar(20)
);

```

```

-- Class
create table Class
(
  class_id varchar(20) primary key,
  class_name varchar(20)
);

-- Booking
create table Booking
(
  booking_id varchar(20) primary key,
  start_date date,
  end_date date,
  start_time varchar(20),
  end_time varchar(20),
  class_id varchar(20),
  room_id varchar(20)
);

-- instructor
create table instructor
(
  instructor_id varchar(20) primary key,
  instructor_name date,
  class_id date
);

```

- a. SQL code used to create the referential integrity constraints;

For the creation of foreign keys, insert into the table. Verifying entities and references is the next step.

- a. SQL code used to create the check constraint(s);

There is a restriction on the area of the gender bar and status bar.

- a. SQL code to build the database performance tuning enhancements.

Database performance can be improved by indexing by improving query efficiency.

- a. SQL code to populate each table with realistic sample data

SQL

```

insert into type values('001','Week Day ');

```

```
insert into member values('001','xiaoming', 'shanghai
province pudong distribute','13370279268','2022-10-
01','2022-11-01','233','001','abc');
insert into class values('abc','Squash') ;
insert into booking values('20221001','2023-01-05','2023-02-
05','2023-01-05 10:00:00','2023-02-05
10:00:00','abc','123');
insert into instructor values('202212','shenglei','abc');
```

a. SQL code to demonstrate evidence of testing carried out to validate the constraints

You can verify entity integrity, reference integrity, user-defined constraints, and so on, by adding, deleting, modifying, and querying.

a. SQL code to demonstrate evidence of transaction rollback and commit operations

After executing the nonconforming insert statement, perform a rollback.

The second step is to delete the incorrect statement and resubmit it,

The transaction can be explicitly committed after the update statement has been completed

a. SQL code to implement and test functions and triggers.

Ensure the member table and all insert, update, and delete activities are audited.

2. Performance Tuning

In addition to clustered indexes on each table, there are non-clustered indexes on other tables that serve as secondary indexes.

There is a need to create secondary indexes for some common core tables, such as Member and Booking.

SQL

```
create index m_id on Member(member_id);
create index c_id on Class(class_id);
create index rc_id on Booking(room_id,class_id);
create index i_id on instructor(instructor_id);
create index tp_id on type(type_id);
```

3. Transaction

Atomicity refers to the fact that a transaction is an indivisible unit of work, and operations within the transaction either occur or do not occur. In order to understand "A to B transfer", we can use the example of "A to B transfer"

By default, SQL transactions are separate transactions in DBMS, and they are automatically committed. When a transaction is started using start transaction, a code block can be executed.

Consistency means that the integrity constraints of the database are not violated before or after a transaction. The integrity of relational data and the consistency of business logic cannot be destroyed by database transactions.

If A transfers to B, regardless of whether the transfer is successful or not, the total deposit amount remains the same (this illustrates the integrity of database relationship constraints).

Guarantee mechanism (also from two aspects): at the database level, before and after the execution of a transaction, the data will conform to the constraints (unique constraints, foreign key constraints, check constraints, etc.) and trigger settings you set; In addition, the internal data structure of the database (such as a B-tree index or two-way linked list) must be correct. Business consistency is generally guaranteed by developers and can also be transferred to databases.

Concurrently accessed transactions are isolated, and one transaction should not affect the running effect of another.

If multiple transactions manipulate the same data at the same time in a concurrent environment, each transaction has its own complete data space. The changes made by concurrent transactions must be isolated from those made by other concurrent transactions. When a transaction views data updates, its status is either the status before another transaction modifies it or the status after another transaction modifies it. Data in the intermediate status will not be viewed by the transaction.

The best understood feature is persistence, which means that after the transaction is completed, the changes made to the database will be permanently saved and cannot be rolled back. A completed transaction is a permanent part of the system, and its impact on the system will last even if the system fails fatally.

To ensure the ACID characteristics of the transaction log, SQL Server uses Write Ahead Logging (WAL). The data is written to the log before being stored in the database.

Stored procedures are used to manage transactions. As an example, commit a transaction after an update, commit a transaction after a group of operations, and do not insert before conditional rollback.

```
SQL
```

```
--A simple example:
```

```

begin;
INSERT INTO instructor VALUES (32, 'Leno');
-- class_no not exists,ERROR,The first insert statement will not be
inserted successfully either.
commit;

-- test
select * from instructor
begin;
INSERT INTO instructor VALUES (37, 'Lixu', '8252235', 'F');
ROLLBACK;
-- The insert operation has been rolled back

```

4. Evaluation of Design and Implementation

SET TRANSACTION sets the session's properties. The SET SESSION CHARACTERISTICS command sets the default transaction characteristics of subsequent sessions. Individual transactions can override these default values with SET TRANSACTION.

Transaction characteristics include transaction isolation level, transaction access mode (read/write or read-only), and deferrable mode. Additionally, you can select a snapshot, but it can only be used for the current transaction and cannot be used as the default for the session.

In parallel transactions, the isolation level of a transaction determines what data it can see:

COMMITTED READ

Statements can only see the rows submitted before they begin. The default value is this.

READ AGAIN AND AGAIN

A transaction's statements can only see rows submitted before the first query or data modification statement.

A serializable format

Statements in the current transaction can only see rows submitted before the first query or data modification statement. In the event that the read/write mode between concurrent serializable transactions may lead to an impossible situation when those transactions are executed serially (one at a time), one of them will be rolled back and a failure error will be generated.

```
SQL
SET TRANSACTION transaction_mode [, ...]
SET TRANSACTION SNAPSHOT snapshot_id
SET SESSION CHARACTERISTICS AS TRANSACTION transaction_mode [, ...]

-- Where transaction_Mode is one of the following:
    ISOLATION LEVEL { SERIALIZABLE | REPEATABLE READ | READ
COMMITTED | READ UNCOMMITTED }
    READ WRITE | READ ONLY
    [ NOT ] DEFERRABLE
```

Export the snapshot from the existing transaction before starting a new transaction. You will receive the snapshot identifier, for example:

```
SQL
BEGIN TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SELECT pg_export_snapshot();
       pg_export_snapshot
-----
00000003-0000001B-1
(1 row)
```

At the beginning of a new transaction, use the snapshot identifier in a SET TRANSACTION SNAPSHOT command:

```
SQL
BEGIN TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SET TRANSACTION SNAPSHOT '00000003-0000001B-1';
```

FINAL GRADE

GENERAL COMMENTS

35/100

Instructor

Bo dong

Generic Assessment Criteria	Marks available	Marks awarded
1. Engagement with Literature Skills	10	
Mark will be awarded on your references to the literature to discuss the importance of the concepts of rollback, commit, functions, triggers, procedures, transactions and indexes. A good discussion of the concepts but unfortunately no references to back up and confirm your work.	2	
2. Knowledge and Understanding Skills	20	
Guidance specific to this assessment: Suitable Entity Relationship design must be employed and justified.	9	

A good ERD was provided but not justified.		
3. Cognitive and Intellectual Skills	20	
Guidance specific to this assessment: <i>Marks will be based on your ability to analyse the case study problem scenario and propose an appropriate, normalised to 3NF, database schema.</i> Schema was normalised to 3nf, although further normalisation is possible.	12	
4. Practical Application Skills	30	
Guidance specific to this assessment: Marks will be awarded based on your ability to build and deploy the database for the ALC case study evidenced by your submission in SQL File 1 (Task 1, item 1).Note: You may be required to provide a demonstration of the implemented database.	8	

<i>Marks will be awarded based on your ability to analyse the case study problem scenario and provide an appropriate practical solution and testing for the ALC database.</i> normalised to 3NF; appropriate attributes to demonstrate the design structure for assignment purposes; Primary key for each table, but no composite primary key No referential integrity constraints (foreign keys); No check constraints. No functions no trigger or function for an auditing feature; no stored procedures a basic transaction not in a procedure selection of Indexes.		
5. Transferable Skills for Life and Professional Practice	20	
<i>Guidance specific to this assessment:</i>	4	

Marks will be awarded based on the written and presentational quality of the report. This will be assessed based on the: · Readability of the report. · Written quality, including spelling and grammar and the use of appropriate formal language. · Structure and presentation of the report. · Critical evaluation of the selected database methods, tools and techniques and their deployment in a multi-user database environment. Use of first person should be avoided in academic writing In future please use numeric referencing format as per university guidelines Overall a well presented and coherent report that would have benefited from figures, and more engagement with literature		
--	--	--

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9
