Database

Data storage

File Based Approach

- The data is stored in one or more seperate computer files

e.g: MS Excel

Relational Database

- Is a way of structuring info in tables, rows and columns

e.g: MS Access

Q- What are the limitations of a file-based approach?

- Data Redundancy (data is repeated in more than one file) e.g: registration data sent twice.
- Data dependancy (changes to data means changes to program accessing the data)
- Lack of Data Integrity (entries that should be same can be different in different places)
- Lack of data privacy (All users have access to all data if a single flat file)

Q- What are the benefits of relational database instead of flat file?

- Reduced data redundancy
- Reduced data dependancy
- Improved data integrity
- Improved data privacy
- Program- data independance
- Ability to create ad hoc queries (queries designed for a particular purpose)

queries → Request for data results / action on data / both

Q- Describe the features of relational database that address the limitations of file based approach.

- Multiple tables are linked together :
- which reduces data redundancy
- increases data integrity
- referential integrity can be enforced

- Program - data independance means that:
- Structure of data can change and can not affect the program
- Structure of programs can change and does not affect data.

- Complex queries can be more easily written
- to find specific data.

- Different users can be given different access rights:
- which improves security

- Different users can be given different views of data:
- so they do not see confidential information
- and data privacy is maintained.

Database Terms

- DBMS: Database Management System (MS Access)

Student table

Roll No.	Name	Course
CS08	Steive	Comp. Sci.
EE54	Jhoson	Electronics
B12	Eva	Biology
F32	Jhoson	Finance
M26	Erica	Maths

- Entity → object
- Attributes → Features of entity
- Fields
- Record / Tuple

Entity: The concept or object in the system that we want to model and store information about

Attributes: A data item, represented as a field within a table

Primary key: A unique identifier for each tuple / Record

Foreign key: A field in one table that links to a primary key in another table.

Candidate key AND Secondary key: Set of keys that could uniquely identify a record is known as candidate keys. One of them is used as primary key and rest are secondary keys.

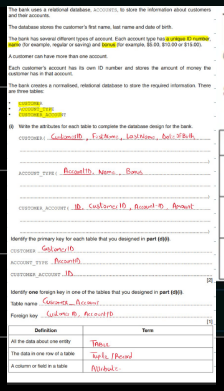
Student table

Roll No

Primary key

Roll No, CNIC, Passport Number, Candidate number

Candidate Key



The bank uses a relational database, ACCOUNTS, to store the information about customers and their accounts.

The database stores the customer's first name, last name and date of birth.

The bank has several different types of account. Each account type has a unique ID number (table) (for example, regular or saving) and bonus (for example, $5.00, $10.00 or $15.00).

A customer can have more than one account.

Each customer's account has its own ID number and stores the amount of money the customer has in that account.

The bank creates a normalised, relational database to store the required information. There are three tables:

- CUSTOMER
- ACCOUNT_TYPE
- CUSTOMER_ACCOUNT

(i) Write the attributes for each table to complete the database design for the bank.

CUSTOMER: CustomerID, FirstName, LastName, DoB, PIN

ACCOUNT_TYPE: AccountID, Name, Bonus

CUSTOMER_ACCOUNT: ID, CustomerID, AccountID, Amount

Identify the primary key for each table that you designed in **part (i)(i)**.

CUSTOMER: CustomerID

ACCOUNT_TYPE: AccountID

CUSTOMER_ACCOUNT: ID

Identify **one** foreign key in one of the tables that you designed in **part (i)(i)**.

Table name: Customer_Account

Foreign key: CustomerID, AccountID

Definition	Term
All the data about one entity	Table
The data in one row of a table	Tuple / Record
A column or field in a table	Attribute

The SOFTWARE_MANAGEMENT database has the following tables:

CUSTOMER_DETAILS(CustomerID, CompanyName, Address1, Address2, City)

SOFTWARE_PURCHASED(SoftwareName, SoftwareDescription, CustomerID, LicenceType, LicenceCost, RenewalDate)

Term	Example
Entity	SOFTWARE_PURCHASED // CUSTOMER_DETAILS.
Foreign key	Customer ID
Attribute	SoftwareDescription.

Primary Key | customer ID

Q- Describe the role of a primary key and Foreign key in a database.

- Primary key uniquely identifies each record
- Primary key can be used as a foreign key in another table
- to form relationships b/w the tables.

Structured Query Language

- Data definition Language (DDL) ⟶ Used for creation of database
- Data Manipulation Language (DML) → Used to manipulate database

Data Definition Language

Q- DDL statement to create a database

```
CREATE DATABASE <Name> ;
```

Q- DDL Statement to create a table

- INTEGER 54
- Date 04/04/2020

- Time (18:00)
- Varchar (string)
- Varchar(n) (string which can contain a maximum of n characters)

- CREATE TABLE < Name > (

  < attribute name > < datatype >,

  < attribute name > < datatype >,

  < attribute name > < datatype >,

  Primary key(name) NOT NULL

);

EmployeeID	FirstName	LastName	DateOfBirth	Gender	DepartmentNumber
156FJEK	Harvey	Kim	12/05/1984	Male	S1
558RRKL	Catriona	Moore	03/03/1978	Female	F2
388LMDV	Oscar	Ciao	01/01/1987	Male	F2

(ii) Write a DDL statement to define the table EMPLOYEE_DATA, and declare EmployeeID as the primary key.

CREATE TABLE EMPLOYEE_DATA (

EmployeeID Varchar(7),

FirstName Varchar ,

LastName Varchar ,

```
    DateOfBirth  Date,
    Gender  Varchar(6),
    DepartmentNumber  Varchar(2),
    PRIMARY KEY(EmployeeID) NOT NULL
);
```

Note: 0 is not considered in integer data type if written like this → 0993212

Q- DDL statement to add new field.

```
ALTER TABLE <table name>
ADD <field name>  <data type>;
```

Data Manipulation Language

Q- Add the data into the table

```
INSERT INTO <table name>
VALUES (<data in column 1>, <data in column 2>, ...);
```

Note: Order is important when adding values in columns.

RoomNumber	RoomType
1	Standard
2	Double
3	Executive
4	Standard

(iii) Room number **5** is a **Double** room.

Complete the Data Manipulation Language (DML) statement to add the details for room number 5 to the table ROOM.

```
INSERT INTO ROOM
VALUES (5, "Double");
```

[2]

It is possible to write the INSERT INTO statement in two ways:

1. Specify both the column names and the values to be inserted:

```
INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);
```

2. If you are adding values for all the columns of the table, you do not need to specify the column names in the SQL query. However, make sure the order of the values is in the same order as the columns in the table. Here, the INSERT INTO syntax would be as follows:

```
INSERT INTO table_name
VALUES (value1, value2, value3, ...);
```

SHOP_SUPPLIER (SHOPID, SUPPLIERID)

(ii) The existing shop with ID 8765 has just used the existing supplier SUP89 for the first time.

Write an SQL script to add this data to the database.

```
INSERT INTO SHOP_SUPPLIER
(SHOPID, SUPPLIERID)
VALUES (8765, 'SUP89');
```

[3]

Q- Update the data / record in a table

```
UPDATE <table Name>
SET <column 1> = <value 1>, <column 2> = <value 2>, ...
WHERE <condition> ;
```

Q- Display / Return a value

```
SELECT <Column 1>, <column 2>, ...
FROM <table name>
WHERE <conditions> ;
```

Table Name = B-Nurse

(ii) Fatima Woo is an Area B nurse with the nurse ID of 076. She has recently married, and her new family name is Chi. Write an SQL command to update her record.

```
UPDATE B-Nurse
SET FamilyName = "Chi"
WHERE NurseID = "076"
```
[3]

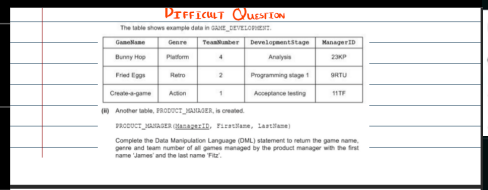
(i) Write an SQL query to display the Nurse ID and family name for all Area B nurses with a specialism of 'THEATRE'.

```
SELECT NurseID, FamilyName
FROM B-Nurse
WHERE Specialism = "THEATRE" ;
```
[3]

LESSON(LessonID, StudentID, InstructorID, LessonDate, LessonTime)

(e) Write a Data Manipulation Language (DML) statement to return the date and time of all future lessons booked with the instructor whose InstructorID is Ins01.

```
SELECT LessonDate, LessonTime
FROM LESSON
WHERE InstructorID = "Ins01"
AND LessonDate > HHHHH ;
```



DIFFICULT QUESTION

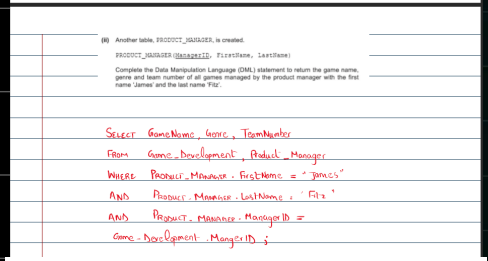
The table shows example data in GAME_DEVELOPMENT:

GameName	Genre	TeamNumber	DevelopmentStage	ManagerID
Bunny Hop	Platform	4	Analysis	23KP
Fried Eggs	Retro	2	Programming stage 1	9RTU
Create-a-game	Action	1	Acceptance testing	11TF

(ii) Another table, PRODUCT_MANAGER, is created.

PRODUCT_MANAGER(ManagerID, FirstName, LastName)

Complete the Data Manipulation Language (DML) statement to return the game name, genre and team number of all games managed by the product manager with the first name 'James' and the last name 'Fitz'.

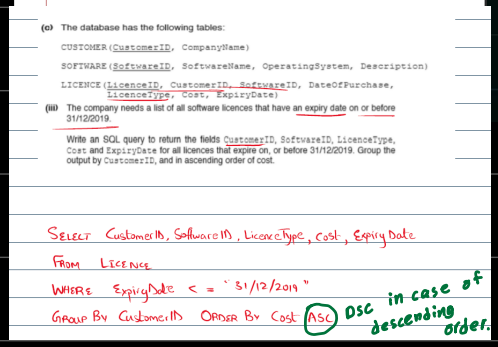


(ii) Another table, PRODUCT_MANAGER, is created.

PRODUCT_MANAGER(ManagerID, FirstName, LastName)

Complete the Data Manipulation Language (DML) statement to return the game name, genre and team number of all games managed by the product manager with the first name 'James' and the last name 'Fitz'.

```
SELECT GameName, Genre, TeamNumber
FROM Game_Development, Product_Manager
WHERE Product_Manager.FirstName = 'James'
AND Product_Manager.LastName = 'Fitz'
AND Product_Manager.ManagerID =
Game_Development.MangerID ;
```



(c) The database has the following tables:

CUSTOMER(CustomerID, CompanyName)

SOFTWARE(SoftwareID, SoftwareName, OperatingSystem, Description)

LICENCE(LicenceID, CustomerID, SoftwareID, DateOfPurchase, LicenceType, Cost, ExpiryDate)

(iii) The company needs a list of all software licences that have an expiry date on or before 31/12/2019.

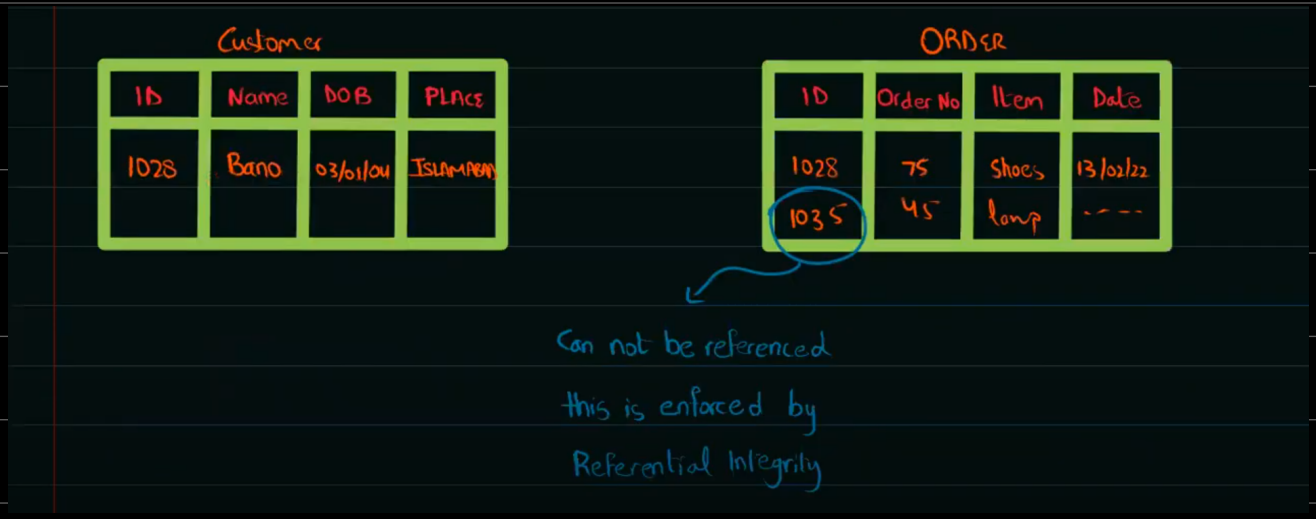
Write an SQL query to return the fields CustomerID, SoftwareID, LicenceType, Cost and ExpiryDate for all licences that expire on, or before 31/12/2019. Group the output by CustomerID, and in ascending order of cost.

```
SELECT CustomerID, SoftwareID, LicenceType, Cost, ExpiryDate
FROM LICENCE
WHERE ExpiryDate <= "31/12/2019"
GROUP BY CustomerID ORDER BY Cost ASC
```

DSC in case of descending order.

Referential Integrity

- Referential Integrity is making sure tables do not try to reference data which does not exist (Foreign key table to base table)
- A primary key can not be deleted unless all dependant records are already deleted.
- Cascading delete (If one data item deleted, the places where it is present are also deleted)
- A primary key can not be updated unless all dependant records are already updated
- Cascading update/edit
- Every foreign key value has a matching value in the corresponding primary key.
- The foreign keys must be the same data type as the corresponding primary key.



Q- What is data redundancy?

- Repeated Data

Q- Explain how a relational database can help to reduce data redundancy?

- Because each record of data is stored once and is referenced by a primary key
- Because data is stored in individual tables.
- and the tables are linked by relationships
- By the proper use of primary and foreign keys.
- By enforcing referential integrity
- By going through the normalisation process.

Q- How Primary key and Foreign key used to link table?

- Name is the primary key in table 1
- links to Name which is the foreign key in table 2

Q- What is data integrity and how you can ensure data integrity?

- Ensure data is consistent
- By enforcing referential integrity, validation rules.

Entity-Relationship Diagrams

- One-to-One — Employee ——— office
- one-to-Many — Teacher ———< Students
- Many-to-one — Students >——— Teacher
- Many-to-Many — Customer >———< Products

Database Management System (DBMS)

Q- Describe the DBMS tools.

Developer Interface:

- To create user friendly features e.g: forms to enter the new booking
- To create outputs e.g: report of bookings on a given table

- To create interactive features e.g: buttons or menus.

Query Processor

- To create SQL queries
- To search for data that meets set criteria e.g: all bookings for next week.
- To perform calculation of extracted data e.g: number of empty rooms
- Organises the results to be displayed.

Q- DBMS has a data dictionary. Describe what data dictionary stores.

- Stores all the information about the database (Metadata)
- For e.g: fields, data types, keys

Q- Tasks performed by DBMS developer interface.

- Create a table
- Set up relationships between tables
- Create a form
- Create a report
- Create a Query *

Q- How DBMS software is used to ensure the security of the data?

Issue usernames and Passwords:

- Stops unauthorised access to data
- Strong password should be used

Access Rights:

- So that only certain usernames can read certain part of the data
- Can be read-only or full access.
- e.g: finance department can edit/read the data related to finance

Create Regular Backups:

- In case of loss/damage to the live data, a copy is available
- e.g: backup at the end of each day.

Encryption of data:

- If there is unauthorised access to the data, it can not be understood.

Normalisation

- Is a method to remove or reduce the repetition of data or redundancy

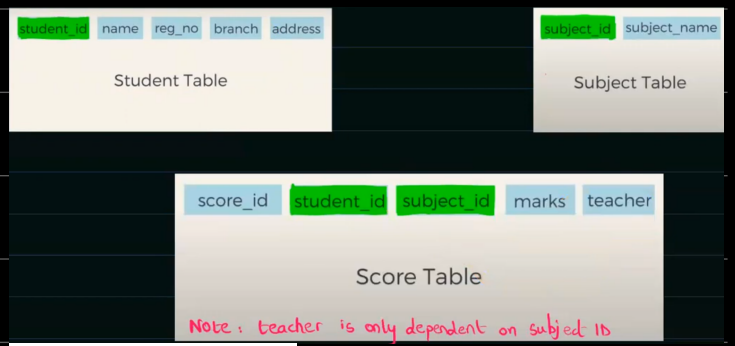
Three stages of Database Normalisation

1 NF :

- No repeated group of attributes
- All attributes should be atomic (Attribute that can not be decomposed into meaningful components)
- No duplicate rows.

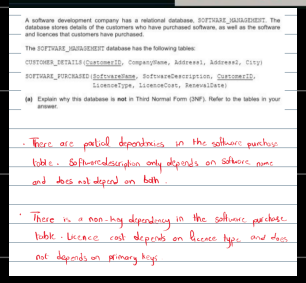
2 NF :

- Should be in 1 NF. (All conditions of 1 NF)
- No Partial dependancy



3 NF :

- Should be in 1 NF and 2 NF (All conditions of 1 NF and 2 NF)
- No non-key / transitive dependancy.


A software development company has a relational database, SOFTWARE_MANAGEMENT. The database stores details of the customers who have purchased software, as well as the software and licences that customers have purchased.

The SOFTWARE_MANAGEMENT database has the following tables:

CUSTOMER_DETAILS(CustomerID, CompanyName, Address1, Address2, City)

SOFTWARE_PURCHASED(SoftwareName, SoftwareDescription, CustomerID, LicenceType, LicenceCost, RenewalDate)

(a) Explain why this database is **not** in Third Normal Form (3NF). Refer to the tables in your answer.

- There are partial dependencies in the software purchase table. SoftwareDescription only depends on software name and does not depend on both.
- There is a non-key dependency in the software purchase table. Licence cost depends on licence type and does not depends on primary keys

Q- Name and describe levels of schema of database.

External Schema:

Schema: Basic structure and format

- The individual's view of database

Conceptual Schema:

- Describes the views which user of the database might have.

Logical Schema:

- Describes how the relationships will be implemented in logic structure of the database

Physical / Internal Schema:

- Describes how the data will be stored on the physical media.