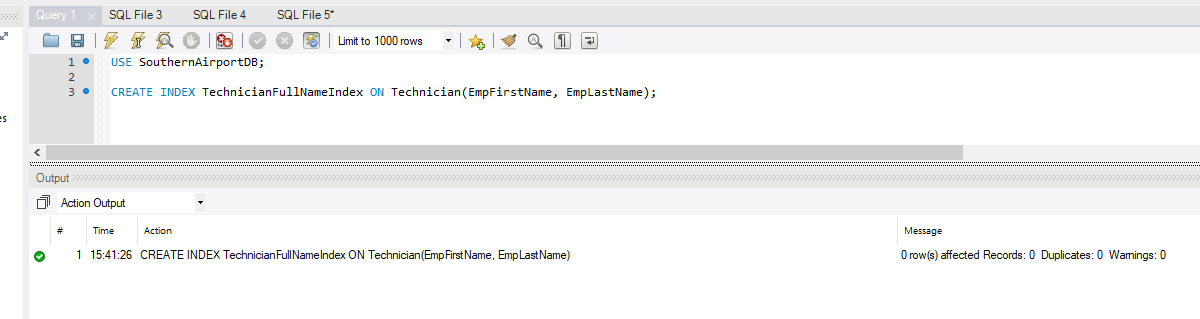
**SECTION C**

(i).

firstname, lastname

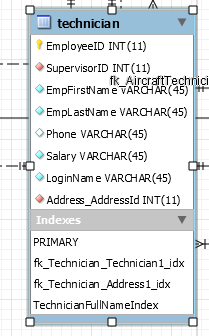
(ii).

CREATE INDEX TechnicianFullNameIndex ON Technician(EmpFirstName, EmpLastName);

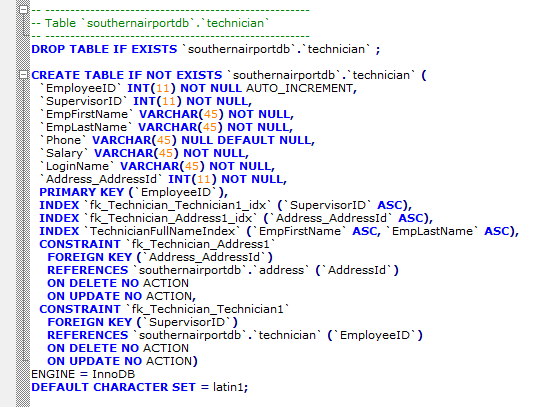


AFTER FORWARD ENGINEER:

The index is in the schema as well.



And in the database script



(i).

Country ( **CountryCode,** CountryName )

Airline ( **IATACode,** *CountryCode,* Website, Phone, Fax, *AddressId* )

Aircraft ( **AircraftId,** *IATACode, PassengerVariantCode, CargoVariantCode,* ModelCode, InternationalRegNo, )

PassengerVariant ( MaxNoPassenger, ***SubModelCode****,* **VariantCode** *)*

CargoVariant ( MaxWeight, ***SubModelCode****,* **VariantCode**)

SubModel ( **SubModelCode,** *ModelCode,* Width, Length, Height, WingSpanArea, MaxCruiseSpeed, MaxRange, TakeOffDist, LandingDist )

SubModel\_has\_EngineModel ( ***SubModelCode, EnginModelCode,*** AircraftNetWeight, TakeOffWeight )

EngineModel ( **EngineModelCode,** EngineWeight )

Engine ( **EngineId,** *EngineModelCode, AircraftId )*

Model ( **ModelCode,** Maker )

ModelTechnician ( ***ModelCode, EmployeeId,*** QualifiedDate*)*

Technician ( **EmployeeId,** *SupervisorId,* EmpFirstName, EmpLastName, Phone, Salary, LoginName, *AddressId* )

Manager ( ***ManagerId***, StartDate)

AircraftTechnician ( ***AircraftId, EmployeeId*** )

TestItem\_has\_TestEvent ( ***TestItemId, TestEventId,*** *EmployeeId, SupervisorId*, TestItemTestEventResult, TestItemTestEventComment )

TestItem ( **ItemCode**, *CASANo,* Description )

Test ( **CASANo,** Description )

TestEvent ( **TestEventId,** *AircraftId, CASANo, PreviousTestEventId,* TestStartTime, TestEndTime, TestResult, Comment )

Address ( **AddressId,** Street, Suburb, PostCode, State, City, Country )

(ii).

|  |  |  |  |
| --- | --- | --- | --- |
| **Table Name** | **Field Name** | **Data Type** | **Relational Description** |
| Aircraft\_Technician | AircraftId  EmployeeId | INT  INT | PK, FK Reference Aircraft (AircraftId)  PK, FK Reference Technician(EmployeeId) |
| TestItem | ItemCode  CASANo  Description | INT  INT  MULTILINESTRING | PK ItemCode,  FK Reference Test(CASANo) |
| Model | ModelCode  Maker | VARCHAR  VARCHAR | PK ModelCode |

(iii).

|  |  |
| --- | --- |
| **DATA DICTIONARY: – ATTRIBUTE DESCRIPTION** | |
| SYSTEM : Southern Airport Maintenance | DATE: 10/05/2019 |
| AUTHOR : Alessandro Ferro | PAGE: 2 of m |
| ATTRIBUTE NAME (CountryCode): Country Code | |
| ALIAS (Synonym) : None | |
| DATA SOURCE: Country | |
| **DATA STRUCTURE** | |
| Type: VARCHAR | |
| Length and Format: AA | |
| Characteristics | |
| Range of Values  Aa ~ Zz | |
| DESCRIPTION  It is a two characters code.  The code already exists outside the database.  It is used to indicate the country of origin of an airline. | |

|  |  |
| --- | --- |
| **DATA DICTIONARY: – ATTRIBUTE DESCRIPTION** | |
| SYSTEM : Southern Airport Maintenance | DATE: 10/05/2019 |
| AUTHOR : Alessandro Ferro | PAGE: 3 of m |
| ATTRIBUTE NAME (VariantCode): Variant identification character | |
| ALIAS (Synonym) : None | |
| DATA SOURCE: CargoVariant, PassengerVariant | |
| **DATA STRUCTURE** | |
| Type: VARCHAR | |
| Length and Format: A | |
| Characteristics | |
| Range of Values  Cc, Pp, Oo | |
| DESCRIPTION  It is a single character.  There are just three possibilities for Cargo, Passenger or Other. | |

|  |  |
| --- | --- |
| **DATA DICTIONARY: – ATTRIBUTE DESCRIPTION** | |
| SYSTEM : Southern Airport Maintenance | DATE: 10/05/2019 |
| AUTHOR : Alessandro Ferro | PAGE: 3 of m |
| ATTRIBUTE NAME (ManagerId): Manager number | |
| ALIAS (Synonym) : None | |
| DATA SOURCE: Manager | |
| **DATA STRUCTURE** | |
| Type: INT | |
| Length and Format: 999999 | |
| Characteristics | |
| Range of Values  000000 ~ 999999 | |
| DESCRIPTION  It is a six digit employee number.  The number is auto increment.  It always has a matching value in the Technician table. | |

(i).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| SOUTHERN AIRPORT MAINTENANCE – AIRCRAFTTEST EVENT SHEET | | | | | | | | | | | | | | | | | | | |
| Airline Code: | | | | |  | | | | | | Aircraft Code: | | | | | |  | | |
| Aircraft Name | | | | | | Aircraft Model | | | | | | | | | Aircraft Sub-Model | | | | |
| *<generated>* | | | | | | *<generated>* | | | | | | | | | *<generated>* | | | | |
| Test Start Date: | | |  | | | | Test Manager ID: | | | | |  | | | | | | Manager Name | |
| *<generated>* | |
| IATA Test Code: | | | | | | | | | | |  | | | | | | | | |
| Item # | Description | | | | | Result | | Technician ID | | | | | Licence No | | | | Date | | Signature |
| *<generated>* | *<generated>* | | | | |  | |  | | | | |  | | | |  | |  |
| *<generated>* | *<generated>* | | | | |  | |  | | | | |  | | | |  | |  |
| *<generated>* | *<generated>* | | | | |  | |  | | | | |  | | | |  | |  |
| *<generated>* | *<generated>* | | | | |  | |  | | | | |  | | | |  | |  |
| Previous Test: | |  | | | | | | | Overall Test Result: | | | | |  | | | | | |
| Comments: | | | | | | | | |
|  | | | | | | | | |
| Date: | | | |  | | | | | | Manager Signature: | | | | | |  | | | |

**SECTION D**

1. Database size

* Technician table

((4+47+47+47+47+47+47+8+47+4)+16384)\*100

* (345+16384)\*100
* 16729\*100

= **1,672,900**

* AircraftModel\_Technician

(4+4+3)\*800

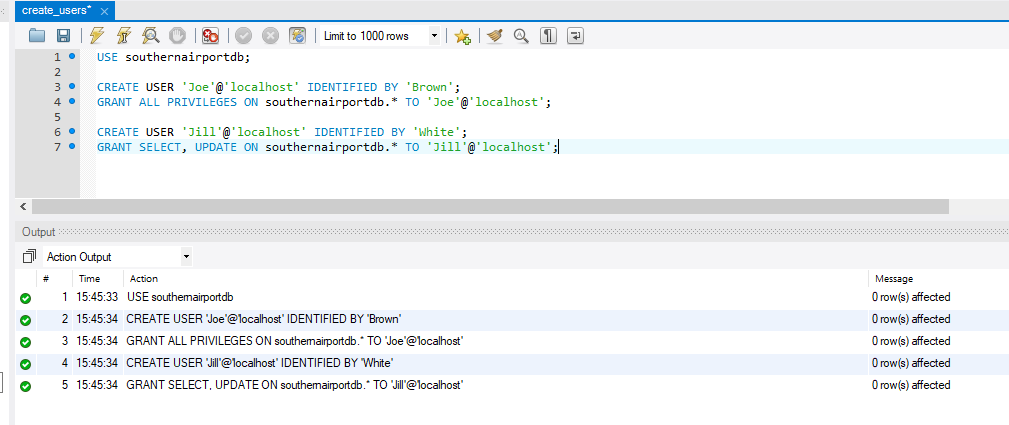
= **8,800**

To calculate the above tables: (field size + index size) \* rows

**SECTION E**

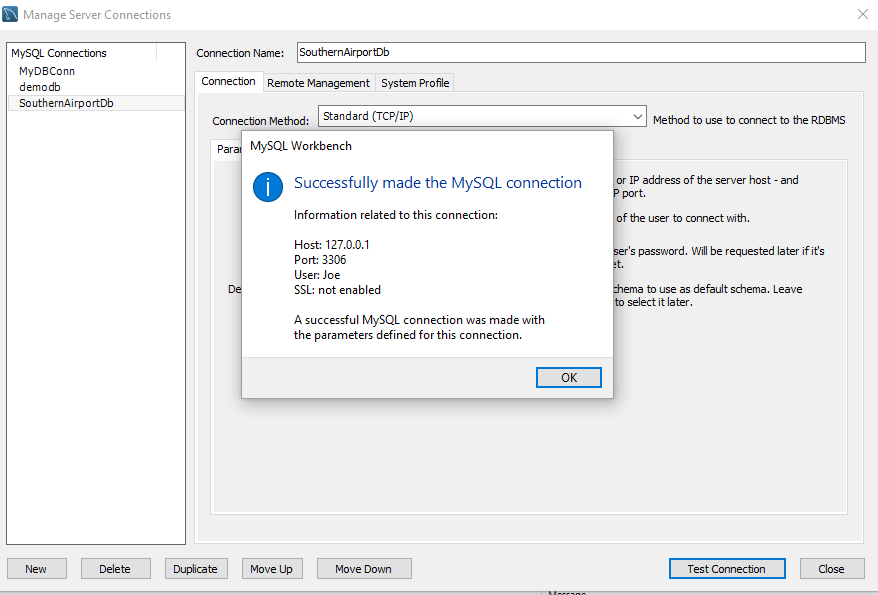
(i).

Create the users and grant them the right privileges.

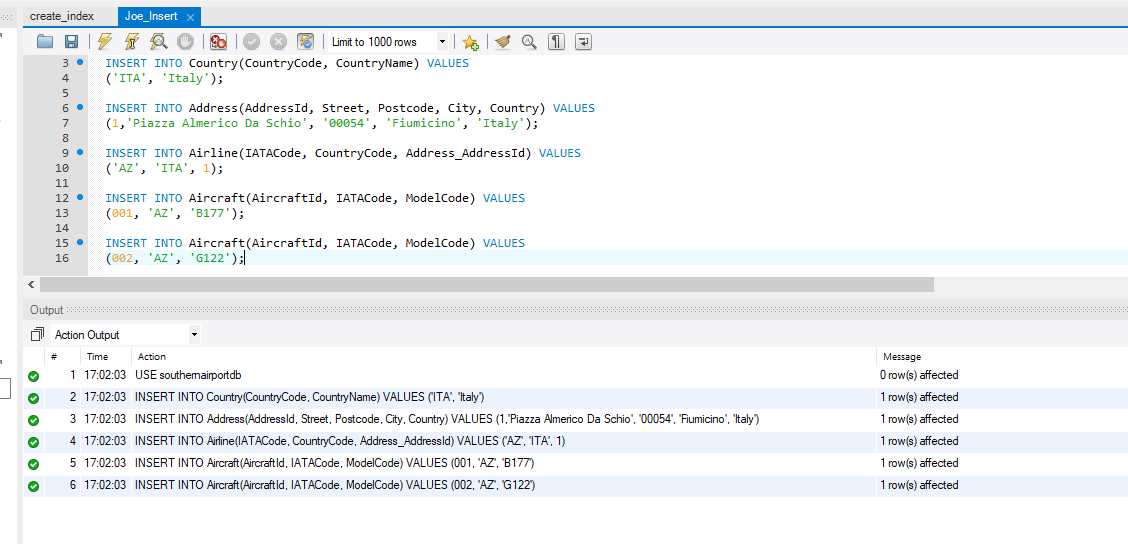


(ii).

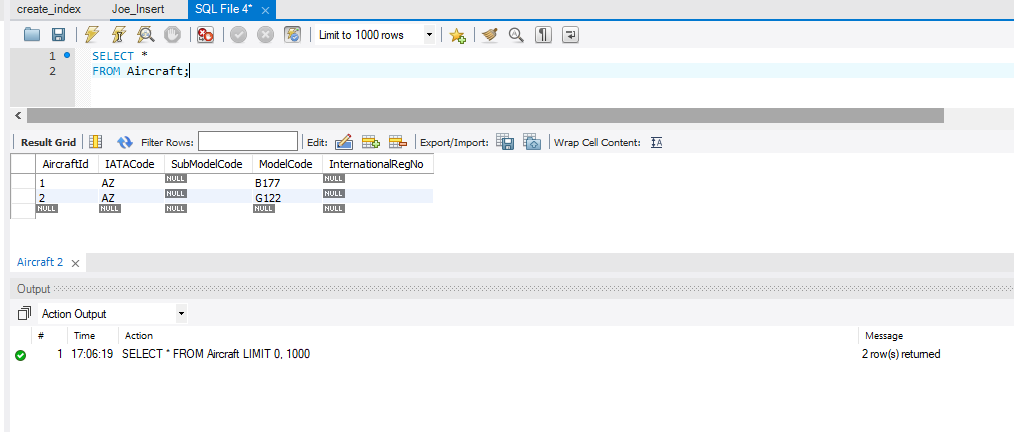
Connect as ‘Joe’ with password ‘Brown’



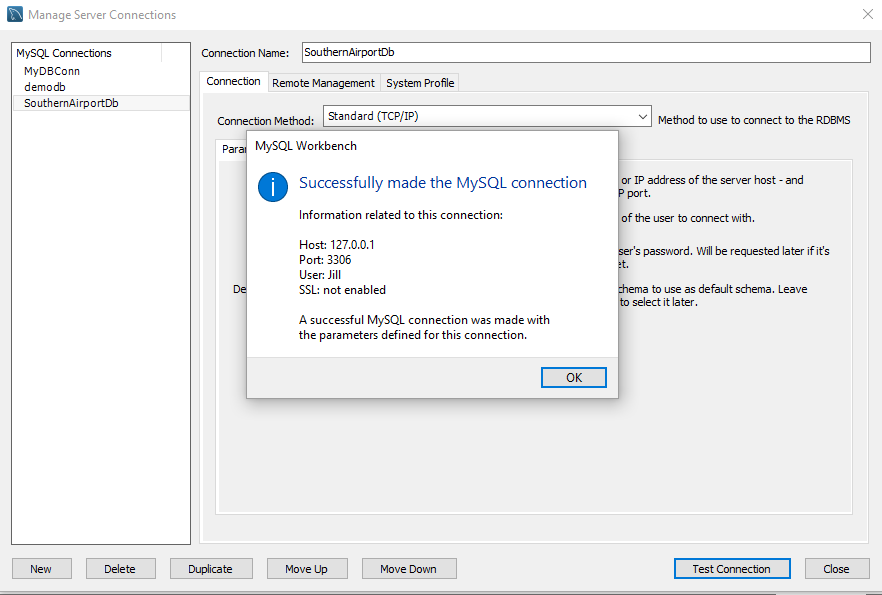
Insert two records in Aircraft table as ‘Joe’.



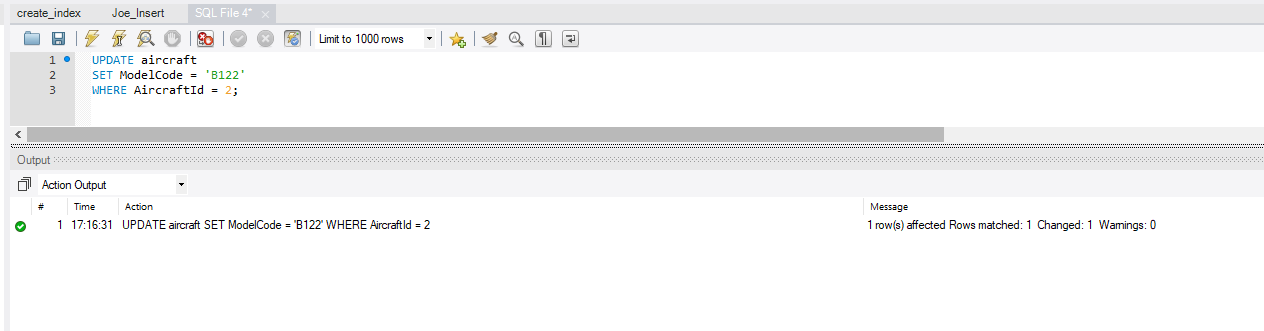
Verify the values are in the table (‘Joe’).



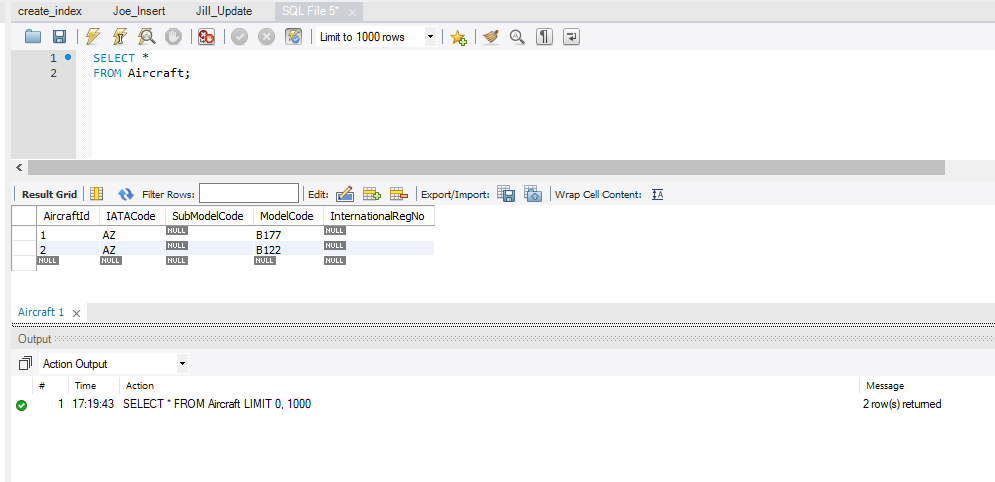
Connect as ‘Jill’



Update aircraft table (‘Jill’)



Verify the table has changed (‘Jill’)



(i).

Encryption is a method to secure data that makes use of algorithm to convert plain text into cipher text. The encrypted data can be accessed only upon decryption.

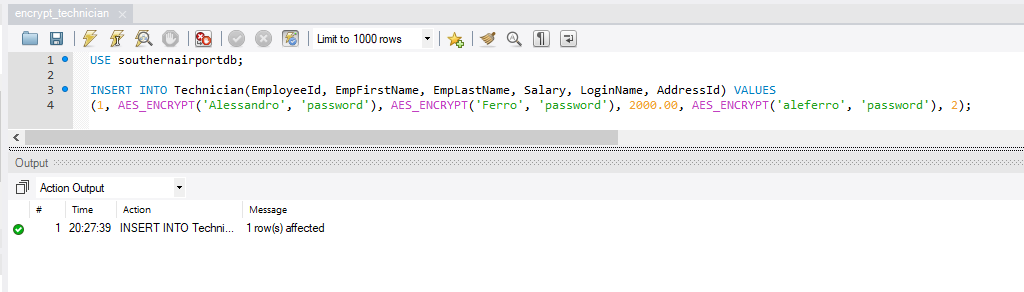
Three common encryption algorithms are:

* DES (Data Encryption Standard): Used to encrypt PINs in ATM machines and to encrypt UNIX passwords. Has replaced by 3DES (or TripleDES), a more secure method that encrypts data three times and uses a different key for at least one of the versions.
* AES (Advanced Encryption Standard): Symmetric secret key algorithm that in the early 2000 replaced DES as the encryption standard for the US Government. The encryption process of this algorithm can be broken down into three different phases, each one with the same sub-operations, but in different combinations.
* SHA1: Cryptographic hashing function that render the encryption as Hexadecimal numbers 40 digits long. Nowadays it is considered unsafe and for sensitive or critical data it is recommended to use a different encryption algorithm such as RSA.

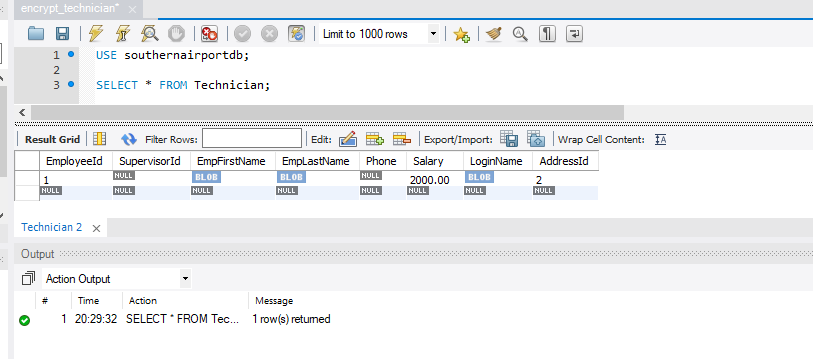
(ii).

I would encrypt the first name, last name and login name of the technician table, the test result and the comment of the test event table. By doing so every query to output those values will return an incomprehensible cipher.

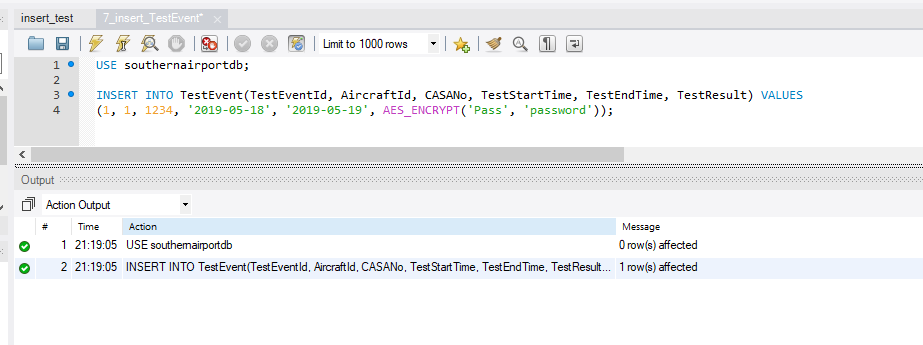
* The Script to encrypt the data in the Technician Table ran successfully.



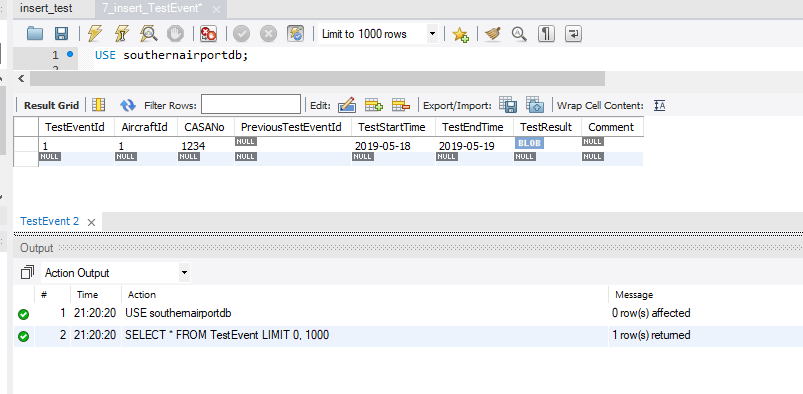
* The result of the encryption after a SELECT query



* The value of TestResult in the TestEvent table has been encrypted successfully.

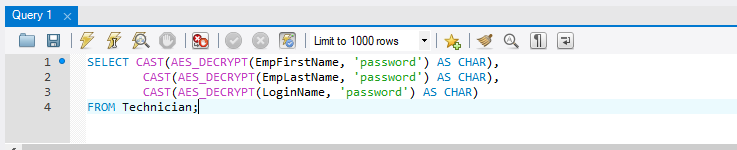


* And the result of a SELECT query return the value encrypted.

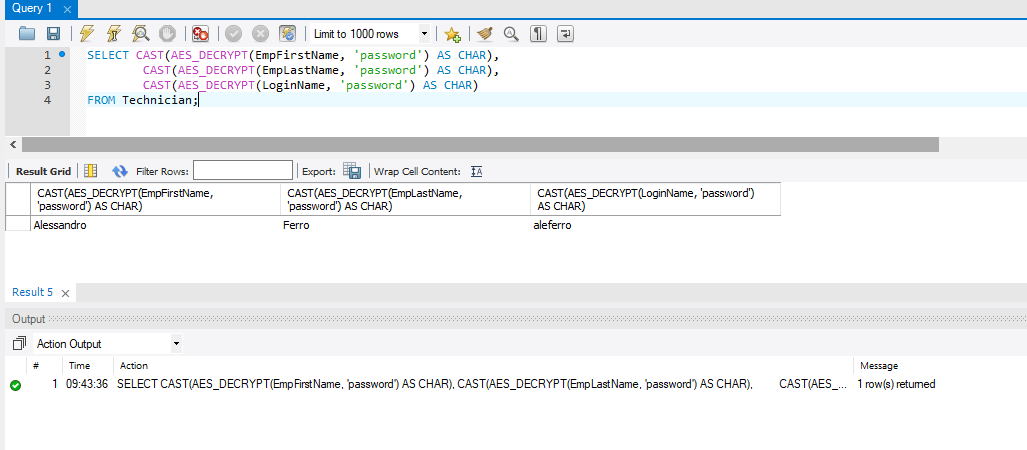


Decryption of the encrypted values in the Technician table

* The query



* The Result



**SECTION F**

**--------------------------------------------------------------------------------------------------------------------------------------**

BACKUP & RECOVERY DOCUMENT – BEGIN

**Introduction**

The purpose of this document is to provide a backup and recovery strategy for the database system of the Southern Airport maintenance operations team.

Following are the reasons why a backup and recovery plan is necessary:

* To safeguard the information assets of Southern Airport Maintenance Operations.
* To prevent the accidental loss of data in case of deletion, system failure or disaster.
* To prevent the corruption of data in case of cyber-attack.
* To quantify the damage that data loss or data corruption would represent for Southern Airport.
* To Delineate a strategy to restore the information in case of data loss or data corruption.

The last session of the document will describe a test plan for this strategy.

**Business Impact**

The work of the maintenance operations team is to conduct tests on the aircrafts in the Southern Airport to verify that the aircrafts are in flight-worthy conditions.

Safety of the data is critical.

The loss of data or the disruption of the service would most likely result in delay in the airport services, with repercussions on the companies and the privates passengers that use the airport services.

In case of data corruption, the risk become even bigger. An aircraft may be deemed flight-ready even with major issues and the risk of extreme damage, or even human life loss would increase greatly.

**Critical Data & Assets**

Critical Data & assets are those data and assets that if lost or compromised could result in major issues such as important economical loss, damage to persons and thing or even death.

In relation to the Southern Airport Database I would propose the following as critical data:

* Airline Name
* Aircraft Code
* Test Start Date
* Test Manager
* IATA Code
* Test Item Number
* Test Item Description
* Test item Result
* Test Item Effectuated Date
* Technician ID
* Technician Licence
* Previous Test (If any)
* Overall Test Result

**Backup Location**

In addition to store the backup files in the server in the maintenance office, the backup files should be also stored in a different device and in the cloud*.*

**Backup & Recovery Solutions & Procedures**

Daily incremental backup.

Weekly full back up.

* Back up From the Command Line – using the mysqldump command connects to the server and creates a file that contains all the statement necessary to to re-create the database.

If the database is very big, it is possible to compress it to a gzip file.

* The database can be restored on a different device.
* The database can be extracted with the wizard in mySQL workbench.

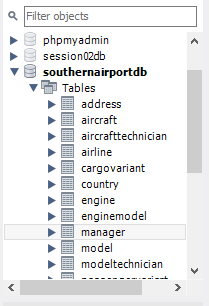
**Test the Plan**

**Approach 1 – The mySQL Workbench wizard:**

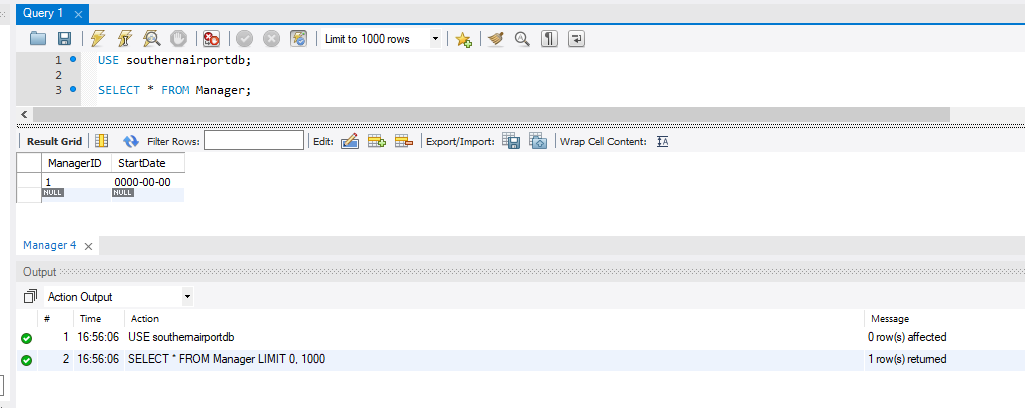
A dump folder can be found in the assignment folder. It contains all the tables extracted using the mySQL Workbench wizard.

**Approach 2 – Creating backup file using the Command Line**

* The manager table is in the database



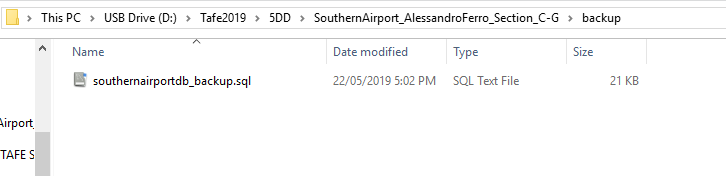
* It contains one record



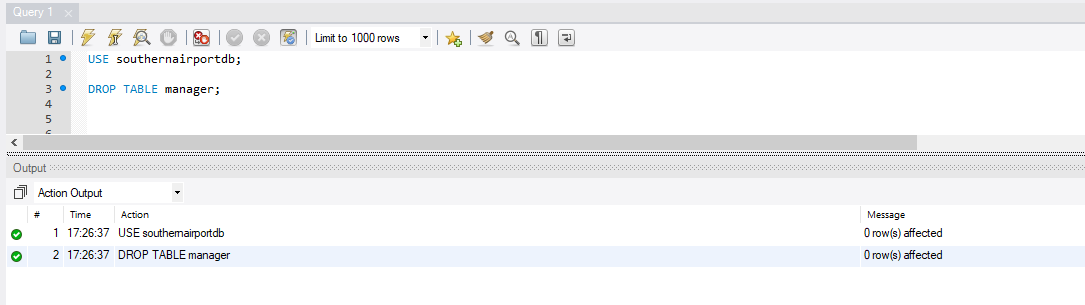
* Run the command to create the backup file

# 

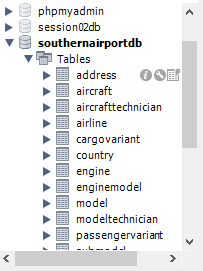
* Backup is created



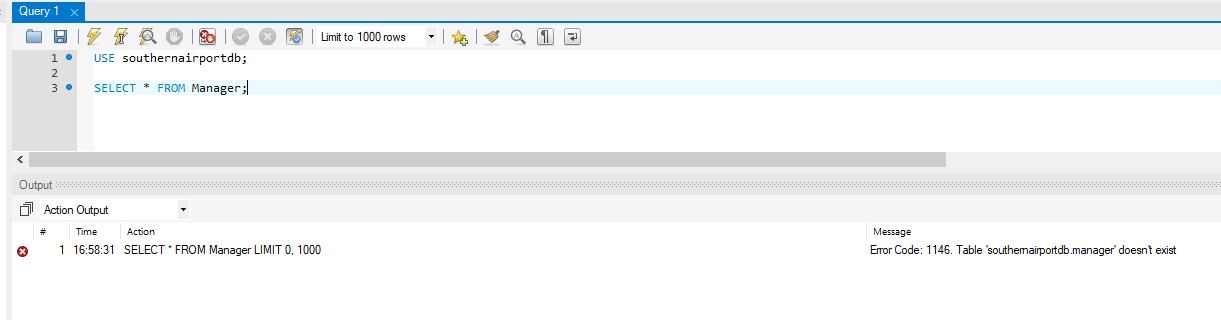
* Drop table Manager



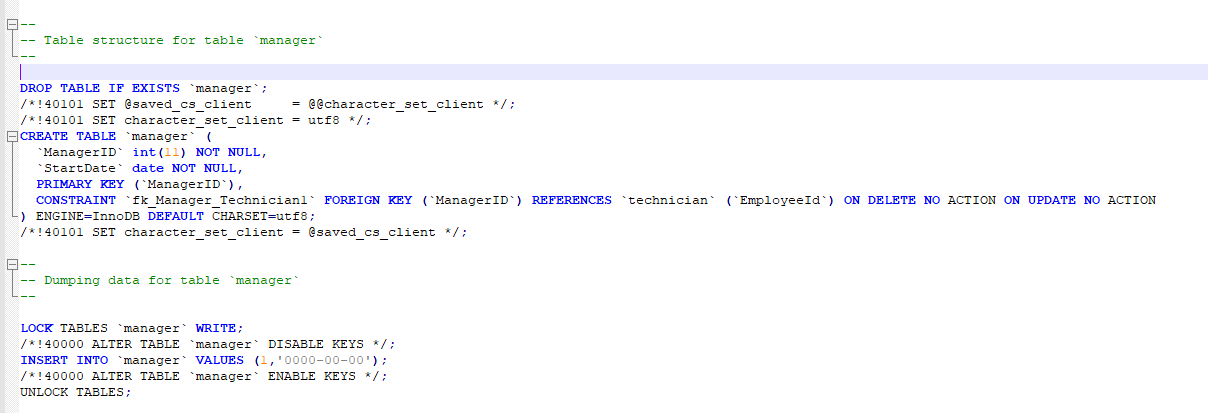
* The table Manager is not in the database anymore



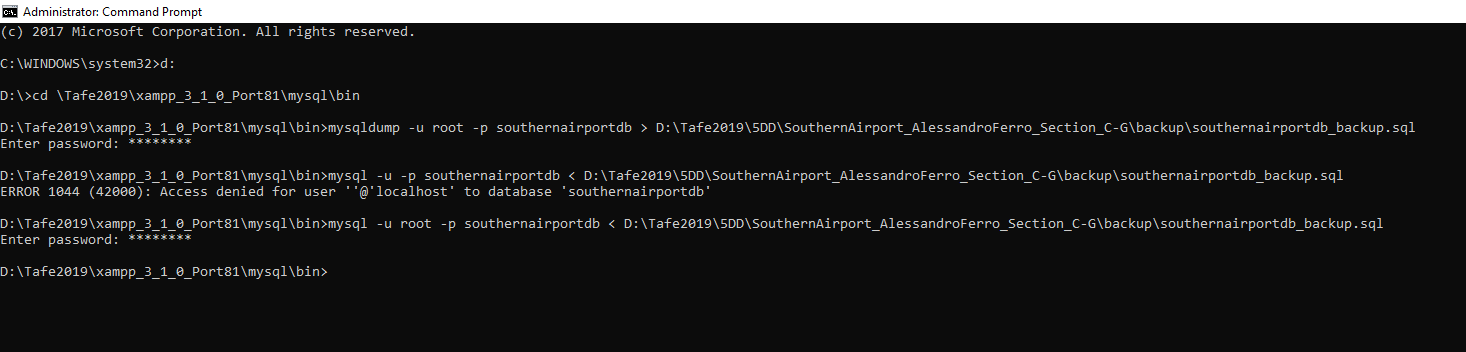
* Running a script to retrieve the data return an error



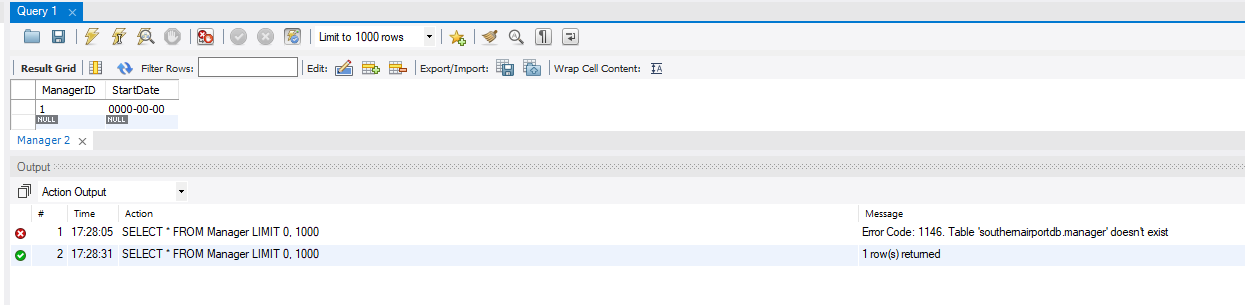
* The table is preserved in the backup file in the dump script.



* Running the script to restore the database



* Running again the script to retrieve the data from the manager table after refreshing the database.



The backup file is available in the assignment folder.

Test plan for the recovery of the database and its tables.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TEST SCOPE | TEST DESCRIPTION | RESOURCES | RESULT EXPECTED | ACTUAL RESULT | FOLLOW UP ACTION RECCOMENDED |
| To test the recovery strategies for the Southern Airport database in case of loss or corruption of data. | A backup file of the database with its data will be generated using the command mysqldump in the command line.  A table in the database will be dropped.  The table will be recovered from the backup file using the command line.  The table health will be put to test with some queries. | O/s wondows with command line  (Run as administrator). | Full recovery of the table with all the data and the constraints as before the test. | The recovery plan proved successful.  Table with all data and constraints was fully restored. | Regular incremental back-up of the database every day and full database back-up once every week. |

Test Plan to Test the table functionalities upon recovery of the Manager table.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Test Command | Test Description | Expected Test Result | Test Resources | Actual Test result | Follow up Action |
| Test 01 | SELECT \* FROM Manager | Retrieve all the data stored in the Manager table as it was before deletion. | Pass | Alessandro Ferro | Pass | None |
| Test 02 | INSERT INTO MANAGER  VALUES  (2, ‘00/00/000’); | Insert an additional record in the Manager table | Pass | Alessandro Ferro | Pass | None |
| Test 03 | UPDATE Manager SET StartDate = 01/01/2001 WHERE ManagerId = 1; | Update one record in the Manager table.  Change the value of one attribute. | Pass | Alessandro Ferro | Pass | None |
| Test 04 | INSERT INTO Manger VALUES (3, ‘11/11/1111’); | Try to insert a record that doesn’t exist in the Technician table. (PK constraint not respected) | Fail | Alessandro Ferro | Fail | None |
| Test 05 | DELETE FROM Manager WHERE ManagerId = 2; | Delete one record from the table | Pass | Alessandro Ferro | Pass | None |

**--------------------------------------------------------------------------------------------------------------------------------------**

BACKUP & RECOVERY DOCUMENT – END

**SECTION G**

During data analysis I analysed the user requirements to identify which information are needed and I used this analysis to create the entities that I used to populate the conceptual model of the database.

Part of this process has been to individuate which data will need to be input by the users to be recorded in the database and which one will be automatically generated by the system, thus allowing for the design of an Input/Output screen.

Follows a list of the pragmatic steps:

* I analysed the business rules to understand which attributes are needed and in which format (data type).
* I identified which attributes would qualify as primary key and I created artefact primary keys when necessary.
* I identified the relationships and the constraint between entities.
* I solved any redundancy, partial dependency and transitivity, following the logical process t go from UNF to 3NF (the minimum normalization level required for a relational database).
* The design of an input screen must keep in consideration which field the users must enter, which can be left empty and which will be automatically generated by the system.
* I analysed the business rules to identify candidates for entities and/or attributes (Nouns).
* I identified the relationships between entities (e.g. 1:1, 1:N, M:N).
* I solved every M:N relationship by creating additional tables.
* I identified entities that where child of other entities.
* I identified all the attributes.
* I identified the primary keys.
* I identified the constraints between tables with the respective foreign-keys.
* I identified which attributes could be obtained from other attributes.

To identify any redundant data in the design of a database it is necessary to undergo the normalization process.

* UNF: Unnormalized Form – Simple data model where multiple values can be stored in the same attribute.
* 1NF: First Normalized Form – Solve repeating groups of data.
* 2NF: Second Normalized Form – Solve partial dependency.
* 3NF: Third Normalized Form – Solve attributes transitivity.

3NF is not the final Normalization Form (7NF is) but is the minimum required to have a functional and efficient database system.

* **Entity Integrity**: Ensure that every entity has a primary key that must be unique and cannot be NULL. The primary key serves as unique identifier for each row in a table and enforces entity integrity by preventing any operation (e.g. INSERT or UPDATE) that would cause to produce a DUPLICATE or invalid primary key.
* **Domain Constraints**: Is defined by data type, length, NULL acceptance, values allowed, default value. Once the attribute of an entity has been defined in all its characteristics, every instance of that attribute must be consistent with those characteristics.
* **Referential Constraints**: If a value exists as Foreign Key in a table, then it must exist as Primary Key in another table or else be NULL. This constraint ensures that any operation (e.g. ADD, UPDATE, DELETE) keeps the relationship between rows in different tables consistent with their primary and foreign keys, or else is not performed.
* **Parental Constraint:** Strictly related with referential integrity. It defines if Cascade Delete and Cascade Update are allowed. If allowed, Delete and Update operations on a parent record influence the rows in the child table with matching Foreign Key.
* **Database authentication:** The database entirely handles the user name and the user password.
* **External authentication:** The user account is maintained by the database, but the password and the authentication are handled by an external service.
  + O/S authentication.
  + Network authentication.
* **Mixed-Mode authentication:** The above can be combined.
* **SSL:** Certificates with a public and private key pair that create an encrypted connection. It is digitally signed by a trusted CA (Certificate Authority).
* **Proxy authentication:** It uses a middle tier application to control the security of the database.
* Database Security Management

. Encryption

. Level-Based Access Control

. Data Masking

* Data Dictionary

. Information about integrity constraints

. Names of schema objects

. Default values

* Database Transaction

. Atomicity

. Consistency

. Isolation

* I analysed the business rules to understand what the data represent.
* I analysed how the user interact with the data.
* I identified the data that need to be entered by the user and the data that can be calculated by the system (derived by other fields).
* I considered the minimum and maximum values for each data to minimise the storage space.
* In screen design, I evaluated which data are relevant to the user, to organize the editable fields and the read-only fields in a way that is meaningful for the user.
* Mapping of entities to tables
* Naming conventions are consistent.
* Inheritance when an entity has child entities.
* The relationships between entities are defined by multiplicity.
* The relationships between entities can be defined by inheritance when necessary (Is a).
* The relationships between tables can be defined by aggregation when necessary (Has a)
* The data entered by the user would become the attributes of each object instance.
* The user interaction with the system will be limited to safeguard data integrity.
* It allows for the use of user defined Abstract Data Types.
* In the screens and report design process, the information gathered would be used to construct objects, thus simplifying data organization and reducing redundancy.
* Considering that the system needs to be up and running even at night time, when the airport is closed, to allow the maintenance team to operate on the aircrafts, this system doesn’t have vertical scalability. However, the system can be scaled horizontally so that it keeps running while being improved.