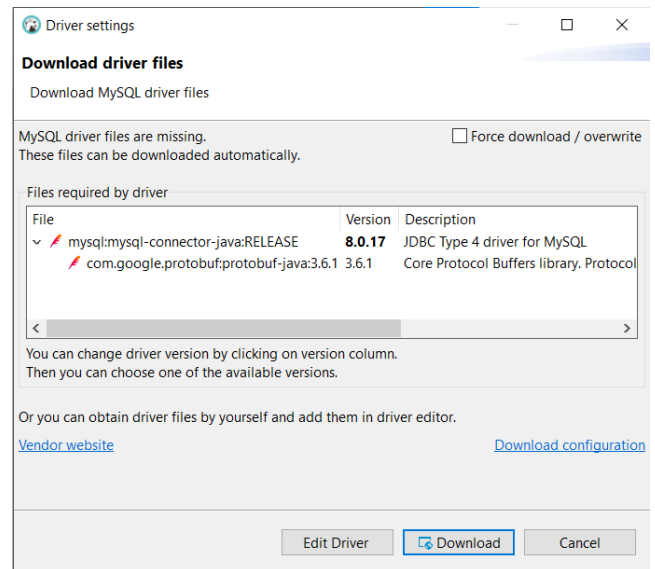
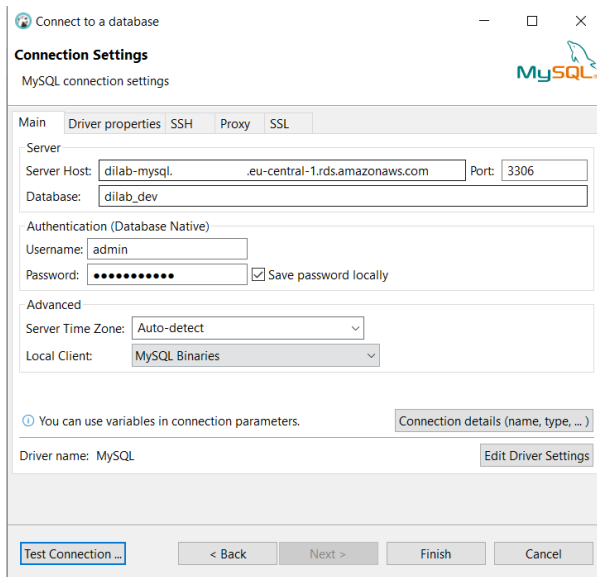


Task_04

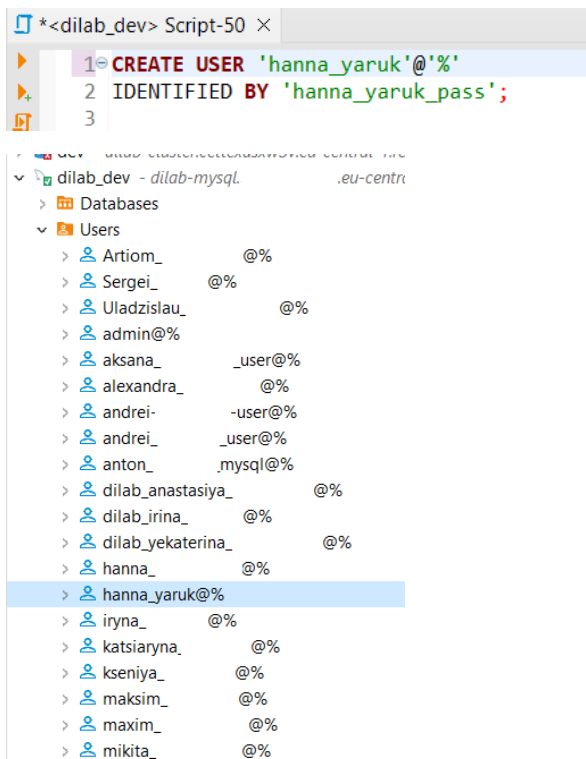
1. RDS MYSQL

Connection to DB through DBvear

First I need to download driver files

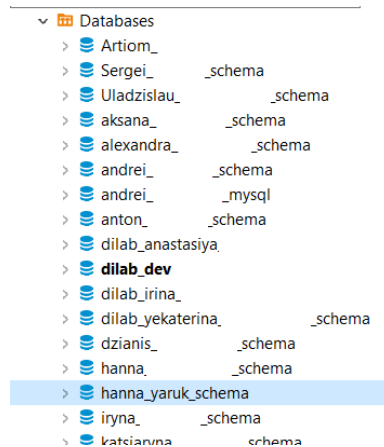


Create a new user



Create my personal schema and grant all privileges to my user

```
5 create schema hanna_yaruk_schema;  
6  
7 GRANT ALL PRIVILEGES ON hanna_yaruk_schema.* TO 'hanna_yaruk'@'%' WITH GRANT OPTION;  
8 FLUSH PRIVILEGES;
```



To use Python to work with MySQL install connector Python MySQL

pip install mysql-connector-python

```
PS C:\Users\Hanna> pip install mysql-connector-python  
Collecting mysql-connector-python  
  Downloading mysql-connector-python-8.0.28-cp310-cp310-win_amd64.whl (7.2 MB)  
    7.2/7.2 MB 5.2 MB/s eta 0:00:00  
Collecting protobuf<=3.0.0  
  Downloading protobuf-3.20.0-cp310-cp310-win_amd64.whl (903 kB)  
    903.8/903.8 kB 7.1 MB/s eta 0:00:00  
Installing collected packages: protobuf, mysql-connector-python  
Successfully installed mysql-connector-python-8.0.28 protobuf-3.20.0  
PS C:\Users\Hanna>
```

Connect to MySQL DataBase with my user

```
task_01.py x qq.py x  
1 import mysql.connector  
2 import sys  
3 import boto3  
4 import os  
5  
6 ENDPOINT="dilab-mysql.          .eu-central-1.rds.amazonaws.com"  
7 PORT="3306"  
8 USER="hanna_yaruk"  
9 PASSWD="hanna_yaruk_pass"  
10 REGION="eu-central-1"  
11 DBNAME="dilab_dev"  
12 os.environ['LIBMYSQL_ENABLE_CLEARTEXT_PLUGIN'] = '1'  
13  
14 #gets the credentials from .aws/credentials  
15 session = boto3.Session(profile_name='default')  
16 client = session.client('rds')  
17  
18 #token = client.generate_db_auth_token(DBHostname=ENDPOINT, Port=PORT, DBUsername=USER, Region=REGION)  
19  
20 try:  
21     conn = mysql.connector.connect(host=ENDPOINT, user=USER, passwd=PASSWD, port=PORT, database=DBNAME, ssl_ca='SSLCERTIFICATE')  
22     cur = conn.cursor()  
23     cur.execute("""SELECT now()""")  
24     query_results = cur.fetchall()  
25     print(query_results)  
26 except Exception as e:  
27     print("Database connection failed due to {}".format(e))
```

```
C:\Users\Hanna\AppData\Local\Programs\Python\Python310\python.exe C:/Users/Hanna/AppData/Roaming/JetBrains/PyCharmCE2021.3/scratches/task_01.py
[(datetime.datetime(2022, 4, 6, 15, 39, 17),)]
Process finished with exit code 0
```

Create table products. Table is created

```
#create tables for the database
create_products_query = """
CREATE TABLE IF NOT EXISTS products(
    id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(100),
    category VARCHAR(100),
    price INT
);
"""
with conn.cursor() as cursor:
    cursor.execute(create_products_query)
    conn.commit()
```

Truncate data from the table before inserting (for restarting). Insert data.

```
#truncate table before inserting new data
truncate_products_query = """
Truncate table products;
"""
with conn.cursor() as cursor:
    cursor.execute(truncate_products_query)

#insert data into the table
insert_products_query = """
INSERT INTO products (id, title, category, price)
VALUES
    (1, "Cucumber", "Vegetables", 8),
    (2, "Tomato", "Vegetables", 11),
    (3, "Banana", "Fruit", 6),
    (4, "Strawberry", "Berry", 30),
    (5, "Apple", "Fruit", 3),
    (6, "Avocado", "Fruit", 13);
"""
with conn.cursor() as cursor:
    cursor.execute(insert_products_query)
    conn.commit()
```

	id	title	category	price
1	1	Cucumber	Vegetables	8
2	2	Tomato	Vegetables	11
3	3	Banana	Fruit	6
4	4	Strawberry	Berry	30
5	5	Apple	Fruit	3
6	6	Avocado	Fruit	13

Create view

```
#create view
create_view_products_query = """
CREATE OR REPLACE VIEW products_view as
SELECT id, title, price
FROM products
"""
with conn.cursor() as cursor:
    cursor.execute(create_view_products_query)
    conn.commit()
```

	id	title	price
1	1	Cucumber	8
2	2	Tomato	11
3	3	Banana	6
4	4	Strawberry	30
5	5	Apple	3
6	6	Avocado	13

- ▼ hanna_yaruk_schema
 - > Tables
 - ▼ Views
 - > products_view
 - > Indexes
 - > Procedures
 - > Triggers
 - > Events

Create procedure for counting products of a category

```
#create_procedure
create_procedure_query = """
CREATE PROCEDURE prod_list (category_insert varchar(64))
BEGIN
    SELECT COUNT(*) FROM products
    WHERE products.category = category_insert;
END;
"""
with conn.cursor() as cursor:
    cursor.execute(drop_procedure_query)
    cursor.execute(create_procedure_query)
conn.commit()
```

- ▼ hanna_yaruk_schema
 - > Tables
 - ▼ Views
 - > products_view
 - > Indexes
 - ▼ Procedures
 - checkUser
 - prod_list
 - > Triggers
 - > Events

2. RDS AURORA

Connection to the schema Sakila from AWS Console

Connect to database ✕

You need to choose a database and enter the database credentials to use the query editor. We will be storing your credentials and the connection in the AWS Secrets Manager service. [Learn more](#)

Database instance or cluster
dilab-aurora-mysql-cluster ▼

Database username
admin ▼ Delete

Database password

Enter the name of the database or schema - (optional)
sakila

Query statement terminator
; ▼

Cancel Connect to database

Write the query

RDS > Editor: dilab-aurora-mysql-cluster

Editor | Recent | Saved queries

```
1 select title, name, first_name, last_name, description, release_year from film f
2 left join film_category fc on f.film_id = fc.film_id
3 left join category c on c.category_id = fc.category_id
4 left join film_actor fa on fa.film_id = f.film_id
5 left join actor a on fa.actor_id = a.actor_id
6 where length > 60 and rating = 'PG-13'
7 order by title
8 limit 10;
```

Run Save Clear Change database

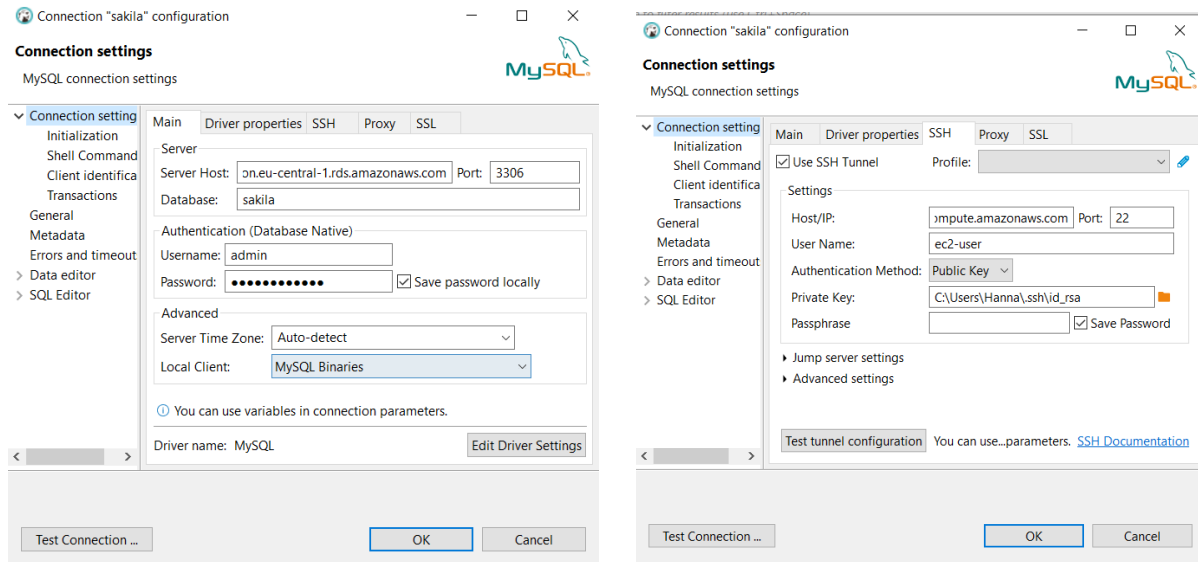
Output Result set 1 (10)

Rows returned (10) Export to csv

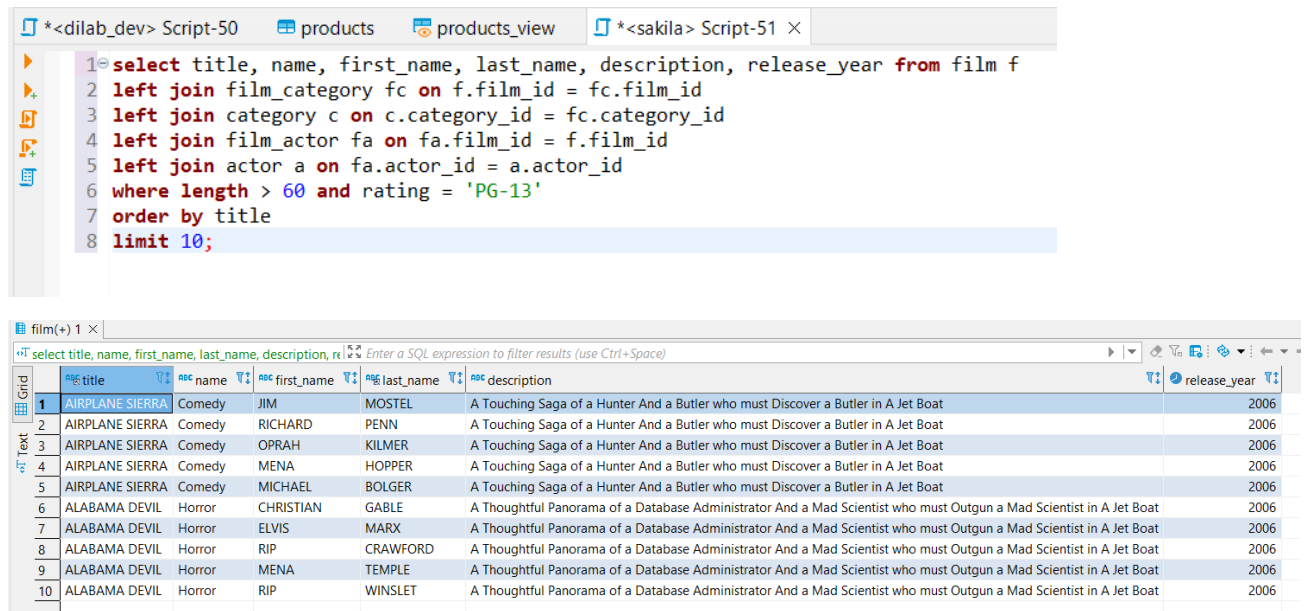
< 1 > ⚙

title	name	first_name	last_name	description	release_year
AIRPLANE SIERRA	Comedy	JIM	MOSTEL	A Touching Saga of a Hunter And a Butler who must Discover a Butler in A Jet Boat	2006-01-01
AIRPLANE SIERRA	Comedy	RICHARD	PENN	A Touching Saga of a Hunter And a Butler who must Discover a Butler in A Jet Boat	2006-01-01
AIRPLANE SIERRA	Comedy	OPRAH	KILMER	A Touching Saga of a Hunter And a Butler who must Discover a Butler in A Jet Boat	2006-01-01
AIRPLANE SIERRA	Comedy	MENA	HOPPER	A Touching Saga of a Hunter And a Butler who must Discover a Butler in A Jet Boat	2006-01-01
AIRPLANE SIERRA	Comedy	MICHAEL	BOLGER	A Touching Saga of a Hunter And a Butler who must Discover a Butler in A Jet Boat	2006-01-01
ALABAMA DEVIL	Horror	CHRISTIAN	GABLE	A Thoughtful Panorama of a Database Administrator And a Mad Scientist who must Outgun a Mad Scientist in A Jet Boat	2006-01-01

Connect to Aurora in DBeaver



Check the connection – use the same select statement



Aurora is serverless service without public IP or public DNS. We cannot work with it from PC. But we have EC2 instance that has public IP. This EC2 is running in same VPC with our Aurora DB cluster, so we should use this EC2 as a jump host for connection with Aurora (or another service in this VPC).

3. DYNAMODB

Create table with AWS Console

Table details [info](#)
DynamoDB is a schemaless database that requires only a table name and a primary key when you create the table.

Table name
This will be used to identify your table.

Between 3 and 255 characters, containing only letters, numbers, underscores (_), hyphens (-), and periods (.).

Partition key
The partition key is part of the table's primary key. It is a hash value that is used to retrieve items from your table and allocate data across hosts for scalability and availability.

1 to 255 characters and case sensitive.

Sort key - optional
You can use a sort key as the second part of a table's primary key. The sort key allows you to sort or search among all items sharing the same partition key.

1 to 255 characters and case sensitive.

Settings

☒ **Default settings**
The fastest way to create your table. You can modify these settings now or after your table has been created.

☐ **Customize settings**
Use these advanced features to make DynamoDB work better for your needs.

Partition Key is Name of wine – it is unique column. Sort key is Region – it helps to select information from the table.

DynamoDB > Tables > hanna_yaruk_wines

Tables (31)

Any table tag

< 1 > ⚙

☐ hanna_lukashevich_book

☒ hanna_yaruk_wines

hanna_yaruk_wines

🔄 Actions ⌵ Explore table items

Overview

Indexes

Monitor

Global tables

Backups

Exports and streams

Additional settings

General information

Partition key Name (String)	Sort key Region (String)	Capacity mode Provisioned	Table status ✔ Active ✔ No active alarms
--------------------------------	-----------------------------	------------------------------	--

▶ Additional info

Items summary

DynamoDB updates the following information approximately every six hours.

Item count
0

Table size
0 bytes

Average item size
0 bytes

Get live item count

Check the existed table with AWS Cli

aws dynamodb describe-table --table-name hanna_yaruk_wines

```
PS C:\Users\Hanna> aws dynamodb describe-table --table-name hanna_yaruk_wines
{
  "Table": {
    "AttributeDefinitions": [
      {
        "AttributeName": "Name",
        "AttributeType": "S"
      },
      {
        "AttributeName": "Region",
        "AttributeType": "S"
      }
    ],
    "TableName": "hanna_yaruk_wines",
    "KeySchema": [
      {
        "AttributeName": "Name",
        "KeyType": "HASH"
      },
      {
        "AttributeName": "Region",
        "KeyType": "RANGE"
      }
    ],
    "TableStatus": "ACTIVE",
    "CreationDateTime": "2022-04-06T22:32:23.411000+03:00",
    "ProvisionedThroughput": {
      "NumberOfDecreasesToday": 0,
      "ReadCapacityUnits": 5,
      "WriteCapacityUnits": 5
    },
    "TableSizeBytes": 0,
    "ItemCount": 0,
    "TableArn": "arn:aws:dynamodb:eu-central-1:260586643565:table/hanna_yaruk_wines",
    "TableId": "5a5d77d5-e989-4e4e-9b07-2929b1beb652"
  }
}

PS C:\Users\Hanna> |
```

Insert the data from wines.json file

aws dynamodb batch-write-item --request-items file://wines.json

```
PS C:\Users\Hanna> aws dynamodb batch-write-item --request-items file://wines.json
{
  "UnprocessedItems": {}
}

PS C:\Users\Hanna> |
```


Check the data in the table from AWS Console

hanna_yaruk_wines

► Scan/Query items
Expand to query or scan items.

Items returned (23)

<input type="checkbox"/>	Name ▾	Region ▾	Colour ▾	Type
<input type="checkbox"/>	Belouve Ro...	Provence	Red	Dry
<input type="checkbox"/>	By.Ott	Provence	Rose	Dry
<input type="checkbox"/>	La Cuvee M...	Languedoc ...	Rose	Dry
<input type="checkbox"/>	Whispering ...	Provence	Rose	Dry
<input type="checkbox"/>	Beaujolais-...	Burgundy	Red	Dry
<input type="checkbox"/>	Sancerre Ro...	Loire Valley	Red	Dry
<input type="checkbox"/>	Chateau de...	Bordo	White	Dry
<input type="checkbox"/>	Clarendelle ...	Bordo	Rose	Dry
<input type="checkbox"/>	Macon-Villa...	Burgundy	White	Dry
<input type="checkbox"/>	Riesling	Alsace	White	Dry

Get the item (unique lines, cannot group)

aws dynamodb get-item --table-name hanna_yaruk_wines --key file:///key.json --return-consumed-capacity TOTAL

```
PS C:\Users\Hanna> aws dynamodb get-item --table-name hanna_yaruk_wines --key file:///key.json --return-consumed-capacity TOTAL
{
  "Item": {
    "Colour": {
      "S": "Rose"
    },
    "Type": {
      "S": "Dry"
    },
    "Region": {
      "S": "Languedoc and Roussillon"
    },
    "Name": {
      "S": "La Cuvee Mythique Rose"
    }
  },
  "ConsumedCapacity": {
    "TableName": "hanna_yaruk_wines",
    "CapacityUnits": 0.5
  }
}
```

PS C:\Users\Hanna> |

Get the select statement – gives the error. Not fixed

aws dynamodb query --table-name hanna_yaruk_wines --key-condition-expression "Name = :n and Region = :r" --expression-attribute-values file://query.json

```
PS C:\Users\Hanna> aws dynamodb query --table-name hanna_yaruk_wines --key-condition-expression "Name = :n and Region = :r" --expression-attribute-values file://query.json
An error occurred (ValidationException) when calling the Query operation: Invalid KeyConditionExpression: Attribute name is a reserved keyword; reserved keyword: Name
PS C:\Users\Hanna> |
```

Delete items

```
PS C:\Users\Hanna> aws dynamodb delete-item --table-name hanna_yaruk_wines --key file://delete_key1.json
PS C:\Users\Hanna> aws dynamodb delete-item --table-name hanna_yaruk_wines --key file://delete_key2.json
PS C:\Users\Hanna> |
```

Items returned (23)

<input type="checkbox"/>	Name	Region	Colour	Type
<input type="checkbox"/>	Riesling	Alsace	White	Dry
<input type="checkbox"/>	Pinot Gris Les Elements	Alsace	White	Semi-dry
<input type="checkbox"/>	Gewurztraminer Jules Geyl	Alsace	White	Sweet
<input type="checkbox"/>	Chateau des Graves Blanc	Bordo	White	Dry
<input type="checkbox"/>	Clarendelle a par Haut-Brion Rose	Bordo	Rose	Dry
<input type="checkbox"/>	Jean-Pierre Moueix Saint-Emilion	Bordo	Red	Dry
<input type="checkbox"/>	Grand Bateau Rouge	Bordo	Red	Dry
<input type="checkbox"/>	Beaujolais-Villages	Burgundy	Red	Dry
<input type="checkbox"/>	Macon-Villages	Burgundy	White	Dry
<input type="checkbox"/>	Bourgogne Pinot Noir La Vignee	Burgundy	Red	Dry
<input type="checkbox"/>	Sauvignon Saint-Bris	Burgundy	White	Dry
<input type="checkbox"/>	La Cuvee Mythique Rose	Languedoc ...	Rose	Dry
<input type="checkbox"/>	Le Grand Noir Grenache-Syrah-Mo...	Languedoc ...	Red	Semi-dry

hanna_yaruk_wines

► Scan/Query items
Expand to query or scan items.

Items returned (21)

<input type="checkbox"/>	Name	Region	Colour	Type
<input type="checkbox"/>	Riesling	Alsace	White	Dry
<input type="checkbox"/>	Pinot Gris Les Elements	Alsace	White	Semi-dry
<input type="checkbox"/>	Gewurztraminer Jules Geyl	Alsace	White	Sweet
<input type="checkbox"/>	Clarendelle a par Haut-Brion ...	Bordo	Rose	Dry
<input type="checkbox"/>	Grand Bateau Rouge	Bordo	Red	Dry
<input type="checkbox"/>	Beaujolais-Villages	Burgundy	Red	Dry
<input type="checkbox"/>	Macon-Villages	Burgundy	White	Dry
<input type="checkbox"/>	Bourgogne Pinot Noir La Vig...	Burgundy	Red	Dry
<input type="checkbox"/>	Sauvignon Saint-Bris	Burgundy	White	Dry
<input type="checkbox"/>	La Cuvee Mythique Rose	Languedoc ...	Rose	Dry