

# ATP 2

```
import org.apache.spark.sql.Session
val spark = Session
    .builder()
    .getOrCreate()
val sqlcontext = spark.sqlContext
sqlcontext
```

FINISHED

```
import org.apache.spark.sql.Session
spark: org.apache.spark.sql.Session = org.apache.spark.sql.Session@263ccfc0
sqlcontext: org.apache.spark.sql.SQLContext = org.apache.spark.sql.SQLContext@7de3901f
res1: org.apache.spark.sql.SQLContext = org.apache.spark.sql.SQLContext@7de3901f
```

Took 25 sec. Last updated by anonymous at September 19 2022, 8:55:08 PM.

```
val sqlcontext = new org.apache.spark.sql.SQLContext(sc)
```

FINISHED

```
warning: there was one deprecation warning; re-run with -deprecation for details
sqlcontext: org.apache.spark.sql.SQLContext = org.apache.spark.sql.SQLContext@31c58606
```

Took 0 sec. Last updated by anonymous at September 19 2022, 8:55:08 PM.

```
val actorVar = sqlcontext
    .read
    .option("header",true)
    .option("delimiter",";")
    .csv("file:/tmp/files/actor.csv")

val addressVar = sqlcontext
    .read
    .option("header",true)
    .option("delimiter",";")
    .csv("file:/tmp/files/address.csv")

val categoryVar = sqlcontext
    .read
    .option("header",true)
    .option("delimiter",";")
    .csv("file:/tmp/files/category.csv")
```

SPARK JOB FINISHED

## ATP 2

```
val cityVar = sqlcontext
    .read
    .option("header",true)
    .option("delimiter",";")
    .csv("file:/tmp/files/city.csv")

val countryVar = sqlcontext
    .read
    .option("header",true)
    .option("delimiter",";")
    .csv("file:/tmp/files/country.csv")

val customerVar = sqlcontext
    .read
    .option("header",true)
    .option("delimiter",";")
    .csv("file:/tmp/files/customer.csv")

val filmVar = sqlcontext
    .read
    .option("header",true)
    .option("delimiter",";")
    .csv("file:/tmp/files/film.csv")

val film_actorVar = sqlcontext
    .read
    .option("header",true)
    .option("delimiter",";")
    .csv("file:/tmp/files/film_actor.csv")

val film_categoryVar = sqlcontext
    .read
    .option("header",true)
    .option("delimiter",";")
    .csv("file:/tmp/files/film_category.csv")

val inventoryVar = sqlcontext
    .read
    .option("header",true)
    .option("delimiter",";")
    .csv("file:/tmp/files/inventory.csv")

val languageVar = sqlcontext
    .read
    .option("header",true)
    .option("delimiter",";")
    .csv("file:/tmp/files/language.csv")
```

```
actorVar: org.apache.spark.sql.DataFrame = [actor_id: string, first_name: string ... 2 more fields]
addressVar: org.apache.spark.sql.DataFrame = [address_id: string, address: string ... 7 more fields]
categoryVar: org.apache.spark.sql.DataFrame = [category_id: string, name: string ... 1 more field]
```

```
cityVar: org.apache.spark.sql.DataFrame = [city_id: string, city: string ... 2 more fields]
countryVar: org.apache.spark.sql.DataFrame = [country_id: string, country: string ... 1 more field]
customerVar: org.apache.spark.sql.DataFrame = [customer_id: string, store_id: string ... 7 more fields]
filmVar: org.apache.spa...
```

# ATP 2

Took 10 sec. Last updated by anonymous at September 19 2022, 8:55:18 PM.

```
actorVar.createOrReplaceTempView("actorTB")
addressVar.createOrReplaceTempView("addressTB")
categoryVar.createOrReplaceTempView("categoryTB")
cityVar.createOrReplaceTempView("cityTB")
countryVar.createOrReplaceTempView("countryTB")
customerVar.createOrReplaceTempView("customerTB")
filmVar.createOrReplaceTempView("filmTB")
film_actorVar.createOrReplaceTempView("film_actorTB")
film_categoryVar.createOrReplaceTempView("film_categoryTB")
inventoryVar.createOrReplaceTempView("inventoryTB")
languageVar.createOrReplaceTempView("languageTB")
```

FINISHED

Took 1 sec. Last updated by anonymous at September 19 2022, 8:55:19 PM.

```
sqlcontext.sql("select * from actorTB").show()
sqlcontext.sql("select * from addressTB").show()
sqlcontext.sql("select * from categoryTB").show()
sqlcontext.sql("select * from cityTB").show()
sqlcontext.sql("select * from countryTB").show()
sqlcontext.sql("select * from customerTB").show()
sqlcontext.sql("select * from filmTB").show()
sqlcontext.sql("select * from film_actorTB").show()
sqlcontext.sql("select * from film_categoryTB").show()
sqlcontext.sql("select * from inventoryTB").show()
sqlcontext.sql("select * from languageTB").show()
```

SPARK JOB FINISHED

```
+-----+-----+-----+-----+
|actor_id|first_name|  last_name|      last_update|
+-----+-----+-----+-----+
|      1| PENELOPE|    GUINNESS|2006-02-15 04:34:33|
|      2|      NICK|   WAHLBERG|2006-02-15 04:34:33|
|      3|       ED|     CHASE|2006-02-15 04:34:33|
|      4| JENNIFER|     DAVIS|2006-02-15 04:34:33|
|      5|   JOHNNY|LOLLOBRIGIDA|2006-02-15 04:34:33|
|      6|     BETTE|  NICHOLSON|2006-02-15 04:34:33|
|      7|     GRACE|    MOSTEL|2006-02-15 04:34:33|
```

ATP 2

8	MATTHEW	JOHANSSON	2006-02-15 04:34:33
9	JOE	SWANK	2006-02-15 04:34:33
10	CHRISTIAN	GABLE	2006-02-15 04:34:33
11	ZERO	CAGE	2006-02-15 04:34:33
12	KARL	BERRY	2006-02-15 04:34:33
13	UMA	WOOD	2006-02-15 04:34:33
14	MTV TEN	BERGEN	2006-02-15 04:34:33

Took 2 sec. Last updated by anonymous at September 19 2022, 8:55:21 PM.

```
sqlcontext.sql("show tables").show();
```

FINISHED

database	tableName	isTemporary
	actortb	true
	addresstb	true
	categorytb	true
	citytb	true
	countrytb	true
	customertb	true
	film_actortb	true
	film_categorytb	true
	filmtb	true
	inventorytb	true
	language	true

Took 1 sec. Last updated by anonymous at September 19 2022, 8:55:22 PM.

```
%spark.sql
select * from actortb;
```

SPARK JOB (<http://587c68636d6c:4040/jobs/job?id=22>) FINISHED

settings

actor_id	first_name	last_name	last_name
1	PENELOPE	GUINNESS	2006

# ATP 2

2	NICK	WAHLBERG	2006
3	ED	CHASE	2006
	JENNIFER	DAVIS	2006
5	JOHNNY	LOLLOBRIGIDA	2006
6	BETTE	NICHOLSON	2006
7	GRACE	MOSTEL	2006

Took 0 sec. Last updated by anonymous at September 19 2022, 8:55:22 PM.

```
actorVar.printSchema()
addressVar.printSchema()
categoryVar.printSchema()
cityVar.printSchema()
countryVar.printSchema()
customerVar.printSchema()
filmVar.printSchema()
film_actorVar.printSchema()
film_categoryVar.printSchema()
inventoryVar.printSchema()
languageVar.printSchema()
```

```
|-- address_id: string (nullable = true)
|-- active: string (nullable = true)
|-- create_date: string (nullable = true)
|-- last_update: string (nullable = true)
```

```
root
|-- film_id: string (nullable = true)
|-- title: string (nullable = true)
|-- description: string (nullable = true)
|-- release_year: string (nullable = true)
|-- language_id: string (nullable = true)
|-- original_language_id: string (nullable = true)
|-- rental_duration: string (nullable = true)
|-- rental_rate: string (nullable = true)
|-- length: string (nullable = true)
|-- replacement_cost: string (nullable = true)
```

FINISHED



```
-- replacement_cost: string (nullable = true)
-- rating: string (nullable = true)
```



Took 1 sec. Last updated by anonymous at September 19 2022, 8:55:23 PM.

## ATP 2

```
%spark.sql
CREATE TABLE ActorVar (
  actor_id BIGINT,
  first_name STRING,
  last_name STRING,
  last_update STRING
) USING orc
```

FINISHED

Took 1 sec. Last updated by anonymous at September 19 2022, 8:55:24 PM.

```
%spark.sql
CREATE TABLE AddressVar (
  address_id BIGINT,
  address STRING,
  address2 STRING,
  district STRING,
  city_id BIGINT,
  postal_code STRING,
  phone STRING,
  location STRING,
  last_update STRING
) USING orc
```

FINISHED

Took 0 sec. Last updated by anonymous at September 19 2022, 8:55:24 PM.

```
%spark.sql
CREATE TABLE CategoryVar (
  category_id BIGINT,
  name STRING,
  last_update STRING
) USING orc
```

FINISHED

Took 0 sec. Last updated by anonymous at September 19 2022, 8:55:24 PM.

```
%spark.sql
CREATE TABLE CityVar (
  city_id BIGINT,
```

FINISHED

```
city STRING,
country_id BIGINT,
last_update STRING
) USING orc
```

Finished

ATP 2

Took 0 sec. Last updated by anonymous at September 19 2022, 8:55:24 PM.

```
%spark.sql
CREATE TABLE countryVar (
  country_id BIGINT,
  country STRING,
  last_update STRING
) USING orc
```

Finished

Took 0 sec. Last updated by anonymous at September 19 2022, 8:55:24 PM.

```
%spark.sql
CREATE TABLE CustomerVar (
  customer_id BIGINT,
  store_id BIGINT,
  first_name STRING,
  last_name STRING,
  email STRING,
  address_id BIGINT,
  active STRING,
  create_date STRING,
  last_update STRING
) USING orc
```

Finished

Took 0 sec. Last updated by anonymous at September 19 2022, 8:55:24 PM.

```
%spark.sql
CREATE TABLE FilmVar (
  film_id BIGINT,
  title STRING,
  description STRING,
  release_year STRING,
  language_id BIGINT,
  original_language_id BIGINT,
  rental_duration STRING,
  rental_rate STRING,
  length STRING,
  replacement_cost STRING,
  rating STRING,
  special_features STRING,
```

Finished

```
last_update STRING
```

Took 0 sec. Last updated by anonymous at September 19 2022, 8:55:25 PM.

## ATP 2

```
%spark.sql
CREATE TABLE Film_actorVar (
  actor_id BIGINT,
  film_id BIGINT,
  last_update STRING
) USING orc
```

FINISHED

Took 0 sec. Last updated by anonymous at September 19 2022, 8:55:25 PM.

```
%spark.sql
CREATE TABLE Film_categoryVar (
  film_id BIGINT,
  category_id BIGINT,
  last_update STRING
) USING orc
```

FINISHED

Took 0 sec. Last updated by anonymous at September 19 2022, 8:55:25 PM.

```
%spark.sql
CREATE TABLE InventoryVar (
  inventory_id BIGINT,
  film_id BIGINT,
  store_id BIGINT,
  last_update STRING
) USING orc
```

FINISHED

Took 0 sec. Last updated by anonymous at September 19 2022, 8:55:25 PM.

```
%spark.sql
CREATE TABLE LanguageVar (
  language_id BIGINT,
  name STRING,
  last_update STRING
) USING orc
```

FINISHED

Took 0 sec. Last updated by anonymous at September 19 2022, 8:55:25 PM.



ATP 2

```
sqlcontext.sql("insert into ActorVar select * from actorTB");
sqlcontext.sql("insert into AddressVar select * from addressTB");
sqlcontext.sql("insert into CategoryVar select * from categoryTB");
sqlcontext.sql("insert into CityVar select * from cityTB");
sqlcontext.sql("insert into CountryVar select * from countryTB");
sqlcontext.sql("insert into CustomerVar select * from customerTB");
sqlcontext.sql("insert into FilmVar select * from filmTB");
sqlcontext.sql("insert into Film_actorVar select * from film_actorTB");
sqlcontext.sql("insert into Film_categoryVar select * from film_categoryTB");
sqlcontext.sql("insert into InventoryVar select * from inventoryTB");
sqlcontext.sql("insert into LanguageVar select * from languageTB");
```

res6: org.apache.spark.sql.DataFrame = []

Took 5 sec. Last updated by anonymous at September 19 2022, 8:55:30 PM.

SPARK JOB FINISHED

```
%spark.sql
select * from FilmVar limit 10
```

SPARK JOB (http://587c68636d6c:4040/jobs/job?id=34) FINISHED



▼

settings ▼

film_id	title	description	release_year	language_id	original_language_id	rent
1	ACADEMY DINOSAUR	A Epic Drama of a Feminist And a Mad Scientist who must Battle a Teacher in The Canadian Rockies	2006	1	0	6
2	ACE GOLDFINGER	A Astounding Epistle of a Database Administrator And a Explorer who must Find a Car in Ancient	2006	1	0	3

Took 1 sec. Last updated by anonymous at September 19 2022, 8:55:31 PM.

ATP 2

```
%spark.sql
select language_id, count(*) as Resultado from FilmVar
GROUP BY language_id
```

SPARK JOB FINISHED

settings ▼

language_id	Resultado
1	1000

Took 2 sec. Last updated by anonymous at September 19 2022, 8:55:33 PM.

```
%spark.sql
select FilmVar.title, LanguageVar.name
from FilmVar join LanguageVar
on LanguageVar.language_id = FilmVar.language_id
```

SPARK JOB FINISHED

settings ▼

title	name
ACADEMY DINOSAUR	English
ACE GOLDFINGER	English
ADAPTATION HOLES	English

ATP 2

AFFAIR PREJUDICE	English
AFRICAN EGG	English
ADONIS TRUMAN	English
AIRPLANE SIERRA	English

Took 1 sec. Last updated by anonymous at September 19 2022, 8:55:34 PM.

```
%spark.sql
select CustomerVar.first_name, CityVar.city from CustomerVar
join AddressVar on AddressVar.address_id = CustomerVar.address_id
join CityVar on CityVar.city_id = AddressVar.city_id
```

SPARK JOB FINISHED

settings ▼

first_name ▼	city
MARY	Sasebo
PATRICIA	San Bernardino
LINDA	Athenai
BARBARA	Myingyan
ELIZABETH	Nantou
JENNIFER	Laredo
MARIA	Kragujevac
SUSAN	Hamilton

Took 1 sec. Last updated by anonymous at September 19 2022, 8:55:35 PM.

# ATP 2

---