

# Intro-Database

## TP-2,3

Université Grenoble Alpes

07.03.2023

[bahareh.afshinpour@univ-grenoble-alpes.fr](mailto:bahareh.afshinpour@univ-grenoble-alpes.fr)

# VPN

## Le VPN

L'utilisation du VPN vous permet à partir de n'importe quel réseau d'accéder aux ressources informatiques du LIG.

Le VPN est désormais proposé par l'Université Grenoble Alpes. La documentation se trouve [ici](#).

Version courte :

- Le VPN de l'Université Grenoble Alpes est ici <https://vpn.grenet.fr>
- Connectez vous en tant que **Personnel UGA**

Vous pouvez aussi utiliser les bastions ssh

# Find your Oracle password

- **Le mot de passe Oracle n'est pas celui de votre compte universitaire.**
- You can find our documentation here (in French): <https://im2ag-wiki.univ-grenoble-alpes.fr/doku.php?id=environnements:oracle>

## Connexion Oracle à partir de septembre 2022

Il faut vous connecter en ssh sur le serveur `im2ag-oracle.univ-grenoble-alpes.fr` avec vos login et mot de passe universitaires :

```
ssh login@im2ag-oracle.univ-grenoble-alpes.fr
```

Ensuite, prenez connaissance de votre mot de passe pour les bases de données Oracle. Il se trouve dans un fichier texte à la racine de votre HOME : `~/oracle.txt`. Votre HOME est monté sur le serveur Oracle, vous pouvez donc utiliser la commande suivante pour afficher votre mot de passe Oracle :

```
cat ~/oracle.txt
```

# To connect to Oracle (DataGrip)

- **DataGrip**
- Please note that DataGrip is not free, but teachers and students of UGA can have the full version for free if you register on their website with your UGA address as a student/teacher account : <https://www.jetbrains.com/datagrip/>
- **hostname:**im2ag-oracle.univ-grenoble-alpes.fr
- **port:**1521
- **servicename:**im2ag

# CREATE TABLE

```
CREATE TABLE table_name (  
    column1 datatype,  
    column2 datatype,  
    column3 datatype,  
    ....  
);
```

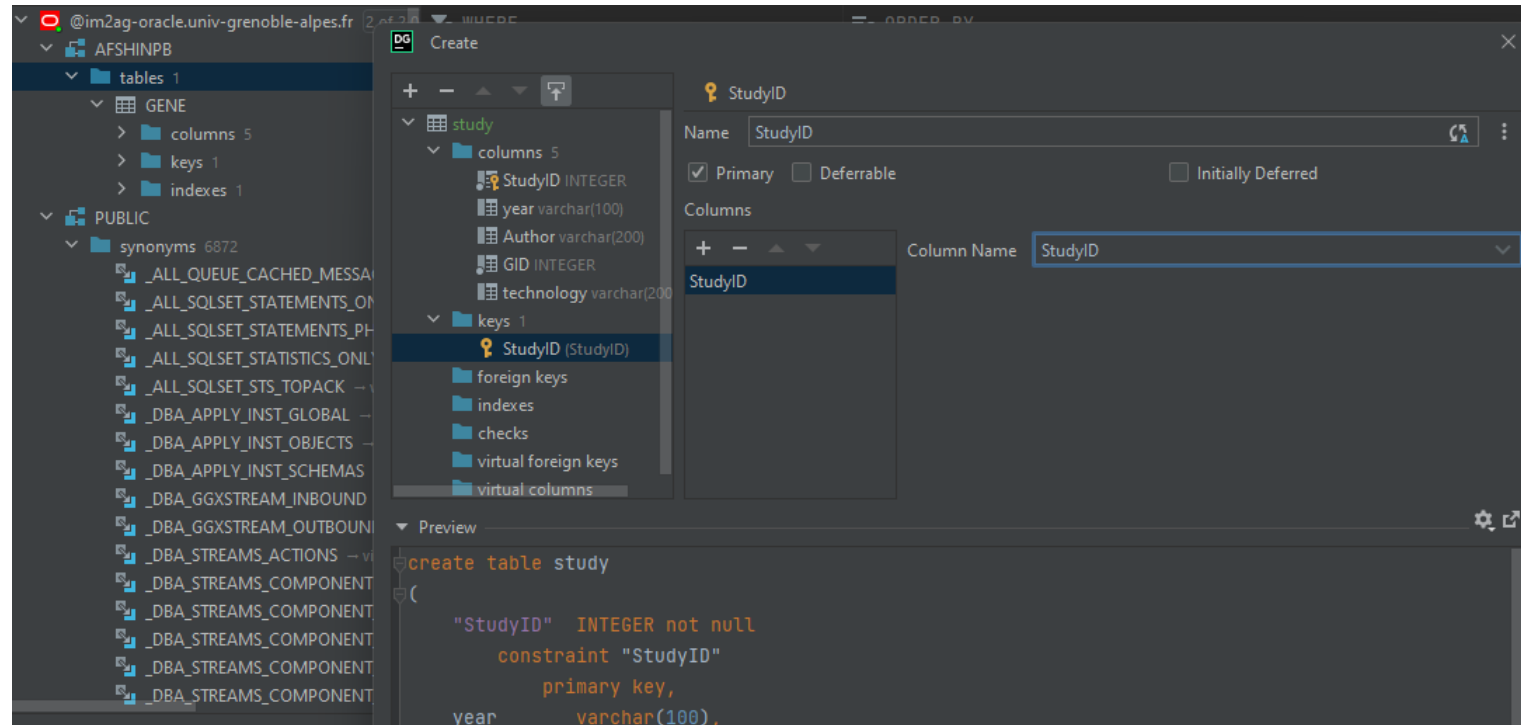
```
CREATE TABLE Persons (  
    ID int NOT NULL PRIMARY KEY,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int  
);
```

A table consists of multiple columns.

It consists of perhaps an ID column, perhaps some kind of a name column, maybe some type of a description column.

# CREATE TABLE

Create table Gene(GId integer NOT NULL  
PRIMARY KEY ,name varchar(100) ,  
symbol varchar(100),s integer,  
chromosome varchar(150) );



The screenshot displays the Oracle SQL Developer interface. On the left, a tree view shows the database structure, including a table named 'GENE' with columns, keys, and indexes. The main window is the 'Create' dialog for a table named 'StudyID'. The dialog shows the table name 'StudyID', a primary key constraint, and a list of columns: StudyID (INTEGER), year (varchar(100)), Author (varchar(200)), GID (INTEGER), and technology (varchar(200)). The 'Preview' pane shows the SQL code for creating the table:

```
create table study
(
  "StudyID" INTEGER not null
    constraint "StudyID"
      primary key,
  year
    varchar(100),
```

# Anatomy of SQL statement

- SQL statement

- INSERT (add a new row in a table)
- UPDATE(modify data in a table)
- DELETE(Remove a row from a table)

These are part of the DML statement.  
DML is data manipulation language.

- Select (you might want to select data or retrieve data from an existing table.)
- FROM clause : It combined with the SELECT clause(statement), would allow you to specify which table or tables you want to retrieve data from.
- WHERE : WHERE clause lets you filter the data that you want to return.

# Insert statement

- You specify insert into,
  - the name of the table.
  - the columns that you want to insert the data into.
  - the keyword values,
  - the data that you want to insert.

The thing is, you have to be careful about how you structure this.

The format, the syntax, the parentheses, the commas, the quotes, the semicolons, everything matters.

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

```
INSERT INTO Customers (Id, CustomerName, Address, City, PostalCode, Country)  
VALUES (50, 'Cardinal', 'Skagen 21', 'Stavanger', '4006', 'Norway');
```

The table name is not case-sensitive.

Without the semicolon, the insert statement would not be processed because Oracle SQL would not be able to understand where the INSERT statement ends.



# Insert statement

```
INSERT INTO Gene(GId,name,symbol,s, chromosome)
values(11998,'tumor protein p53','P53',25.760 , 'chr17');
```

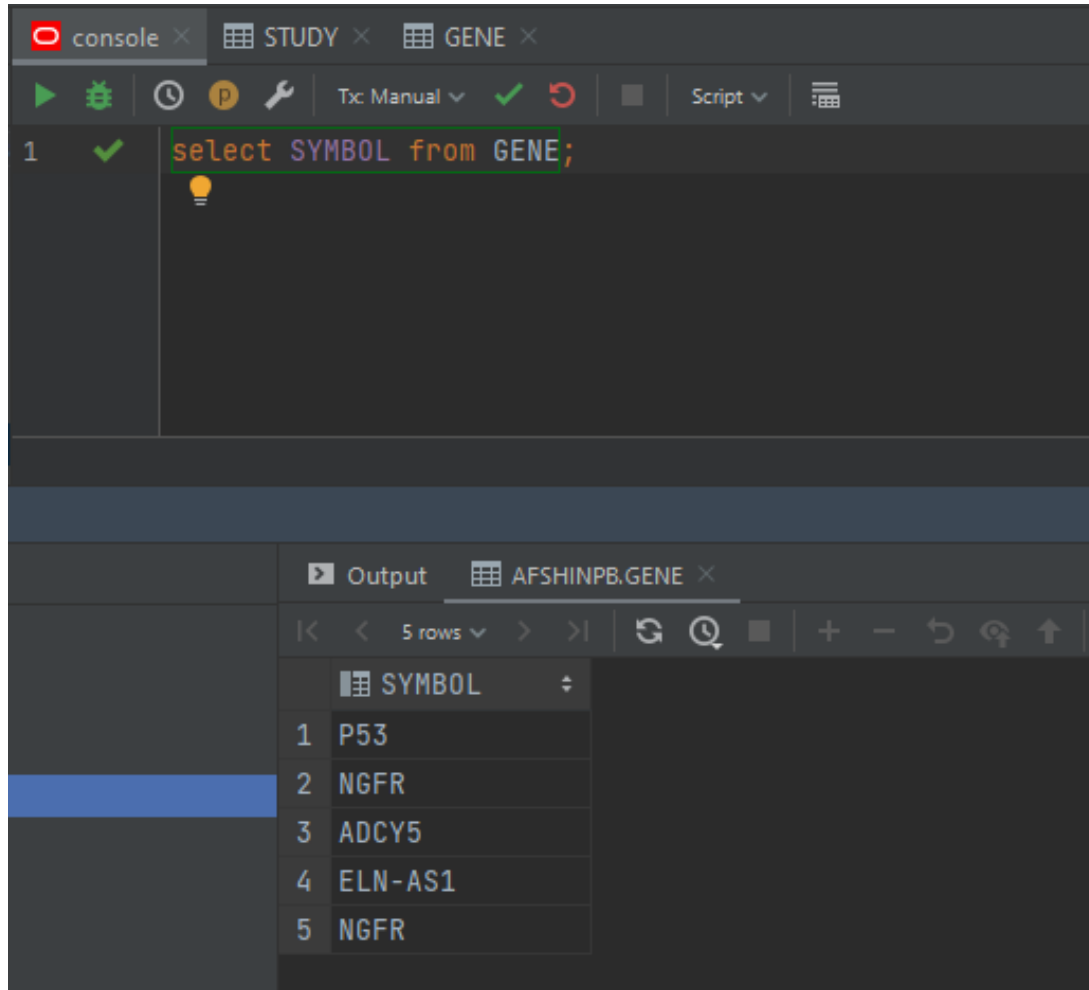
```
insert into Gene(GId,name,symbol,s, chromosome)
values(20856,'ELN Antisense RNA 1','ELN-AS1', 23.98,'chr7');
```



The screenshot shows a database query result with a table containing 5 rows. The columns are GID, NAME, SYMBOL, S, and CHROMOSOME. The rows are ordered by GID. The first row is highlighted in blue.

	GID	NAME	SYMBOL	S	CHROMOSOME
1	11998	tumor protein p53	P53	26	chr17
2	7809	nerve growth factor ...	NGFR	20	chr17
3	1205	ADCY5	ADCY5	30	chr11
4	20856	ELN Antisense RNA 1	ELN-AS1	24	chr7
5	77899	nerve growth factor ...	NGFR	20	chr17

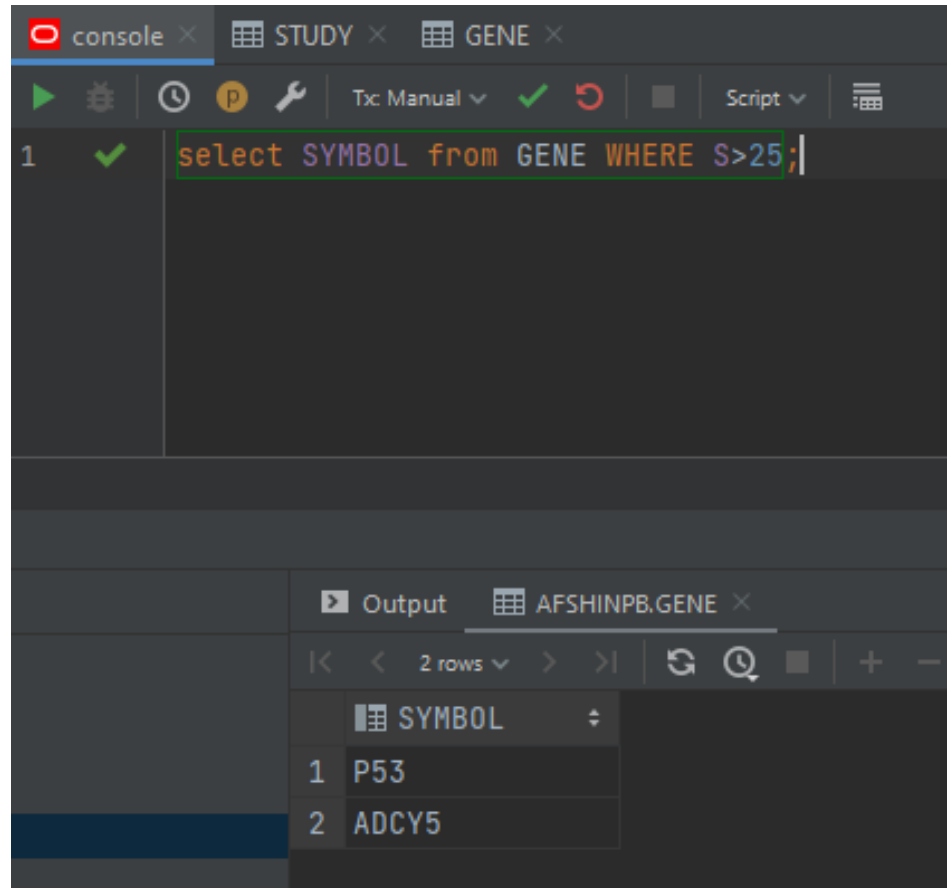
# Give the list of Gene's symbols.



The screenshot shows a database console interface with a dark theme. At the top, there are tabs for 'console', 'STUDY', and 'GENE'. Below the tabs is a toolbar with various icons for execution and settings. The main area contains a single SQL query: `select SYMBOL from GENE;`. Below the query, there is a lightbulb icon indicating a suggestion or tip. At the bottom, the 'Output' window is open, showing the results of the query. The output is a table with one column labeled 'SYMBOL' and five rows of data.

	SYMBOL
1	P53
2	NGFR
3	ADCY5
4	ELN-AS1
5	NGFR

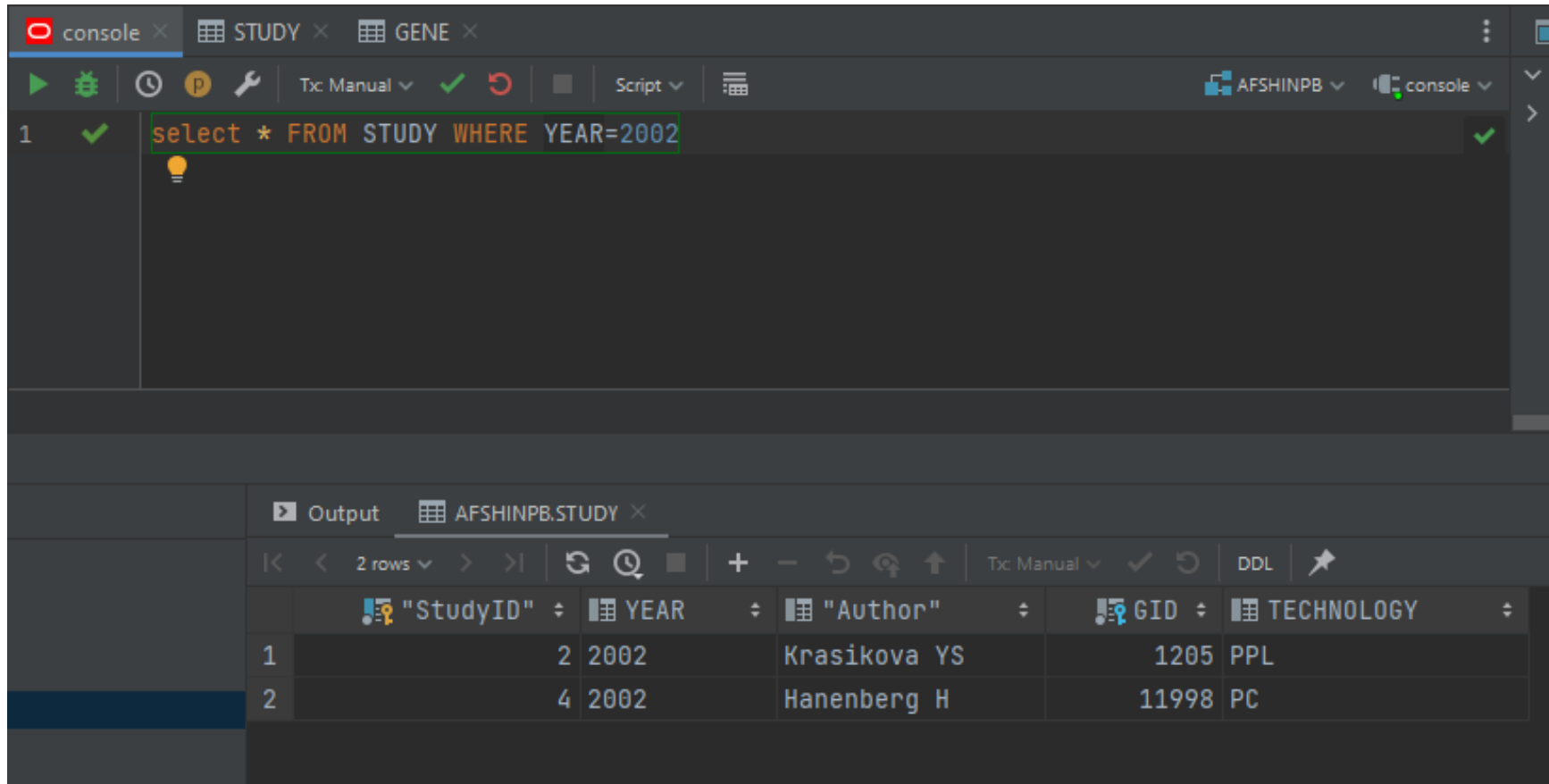
Output the list of genes whose sizes are larger than 25 bases.



The screenshot shows a database console interface with a dark theme. At the top, there are tabs for 'console', 'STUDY', and 'GENE'. Below the tabs is a toolbar with various icons and a 'Script' dropdown menu. The main area contains a single line of SQL code: `select SYMBOL from GENE WHERE S>25;`. Below the code editor, there is an 'Output' section with a tab labeled 'AFSHINPB.GENE'. The output shows two rows of data:

	SYMBOL
1	P53
2	ADCY5

Return the list of authors who studied genes in 2002.

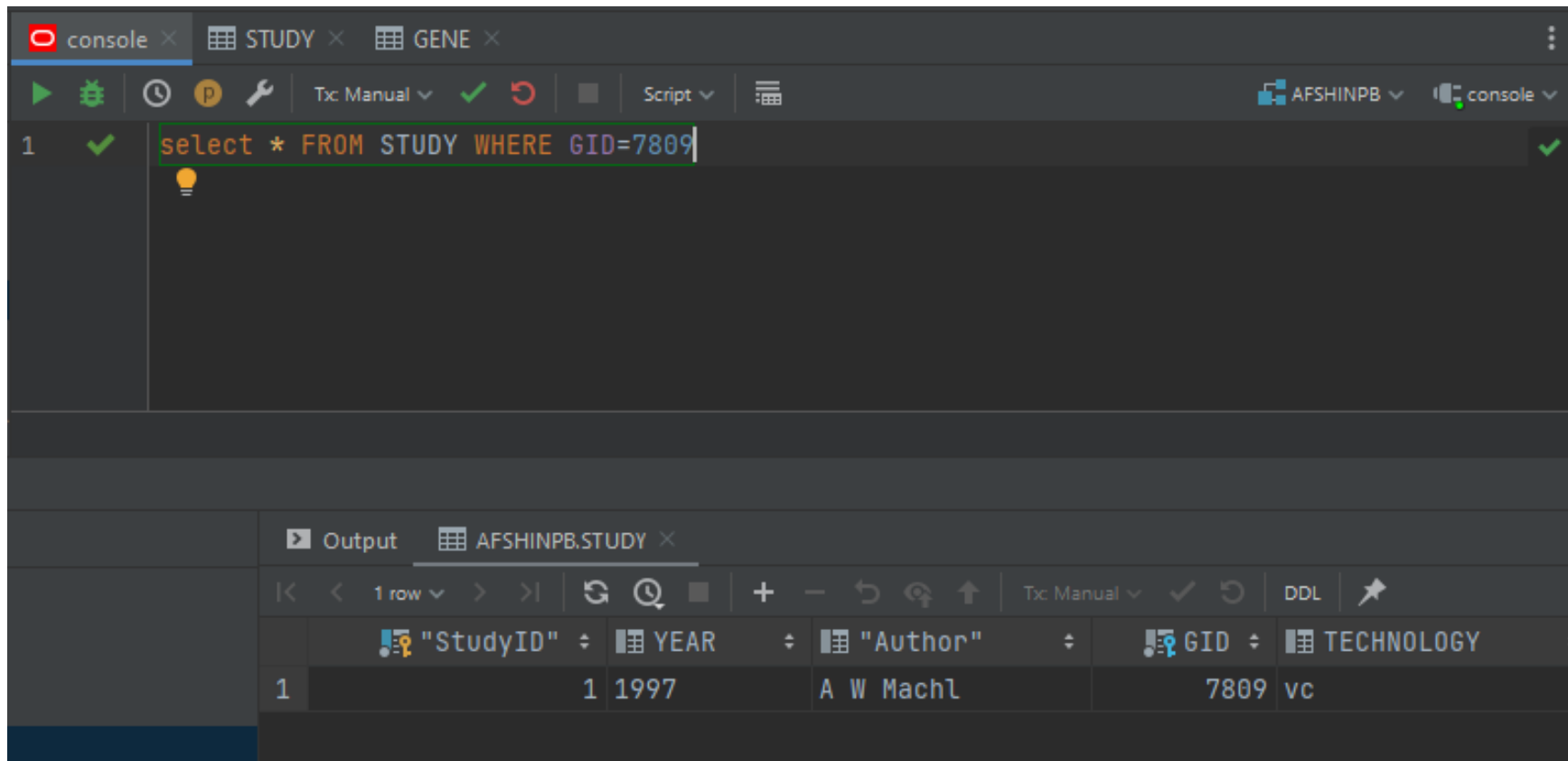


The screenshot shows a database console interface. The top part displays a SQL query: `select * FROM STUDY WHERE YEAR=2002`. Below the query, the output is shown as a table with 2 rows. The table has columns: "StudyID", YEAR, "Author", GID, and TECHNOLOGY. The first row contains the values: 1, 2, Krasikova YS, 1205, PPL. The second row contains the values: 2, 4, Hanenberg H, 11998, PC.

```
select * FROM STUDY WHERE YEAR=2002
```

"StudyID"	YEAR	"Author"	GID	TECHNOLOGY
1	2	Krasikova YS	1205	PPL
2	4	Hanenberg H	11998	PC

# Return the list of authors WHO studied on Gene ID 7809

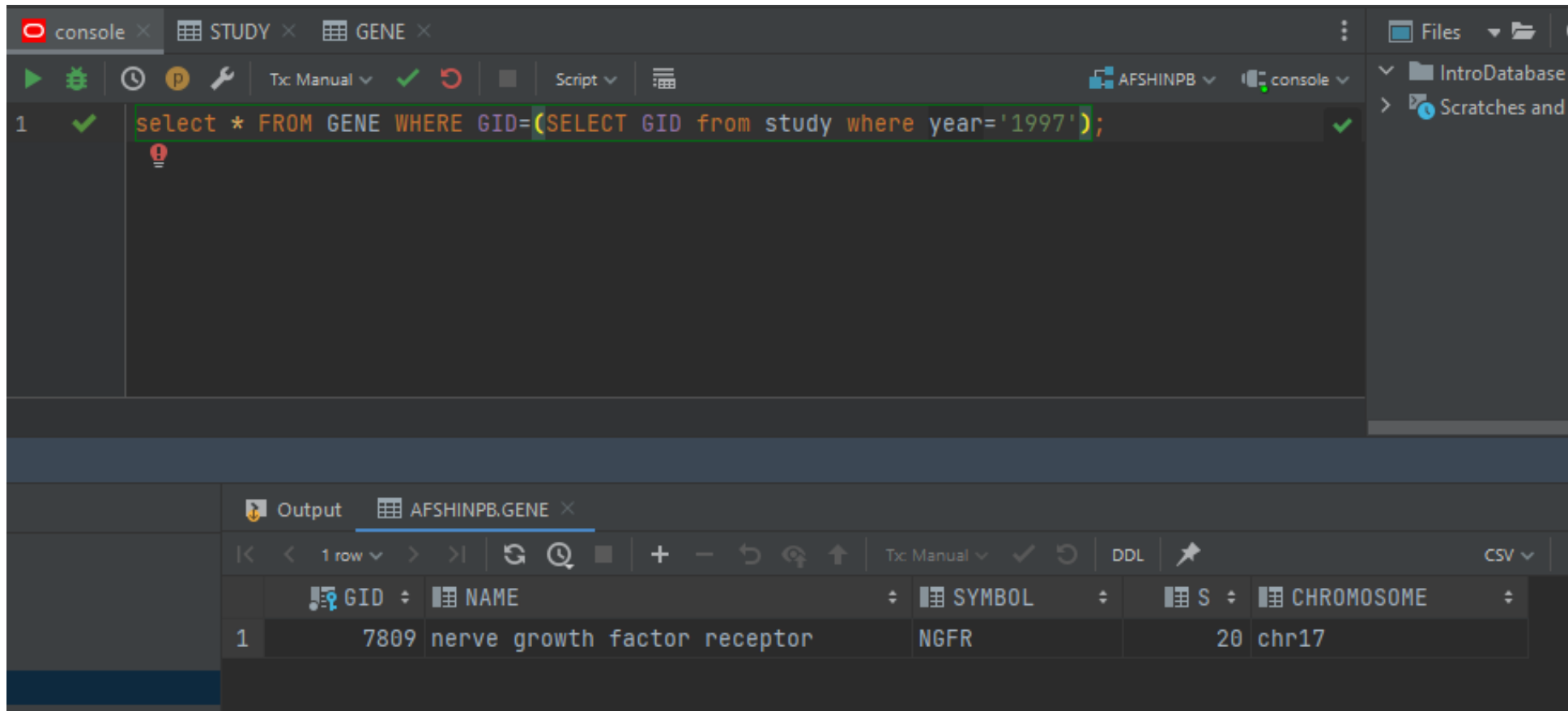


The screenshot shows a database console interface with a dark theme. At the top, there are tabs for 'console', 'STUDY', and 'GENE'. Below the tabs is a toolbar with various icons for execution, refresh, and search. The main area contains a SQL query: `select * FROM STUDY WHERE GID=7809`. Below the query, there is a lightbulb icon indicating a tip or suggestion. The bottom part of the console shows the 'Output' window for 'AFSHINPB.STUDY'. It displays a table with one row of data. The table has columns for 'StudyID', 'YEAR', 'Author', 'GID', and 'TECHNOLOGY'. The data row shows '1', '1997', 'A W Machl', '7809', and 'vc'.

```
select * FROM STUDY WHERE GID=7809
```

"StudyID"	YEAR	"Author"	GID	TECHNOLOGY
1	1997	A W Machl	7809	vc

Output the list of genes that we have information about it in 1997



The screenshot shows a database console interface with a dark theme. At the top, there are tabs for 'console', 'STUDY', and 'GENE'. The console area contains a single SQL query: `select * FROM GENE WHERE GID=(SELECT GID from study where year='1997');`. Below the query, the output is displayed in a table format. The table has five columns: GID, NAME, SYMBOL, S, and CHROMOSOME. The first row of data shows GID 7809, NAME 'nerve growth factor receptor', SYMBOL 'NGFR', S '20', and CHROMOSOME 'chr17'.

```
select * FROM GENE WHERE GID=(SELECT GID from study where year='1997');
```

GID	NAME	SYMBOL	S	CHROMOSOME
1	7809	nerve growth factor receptor	NGFR	20 chr17

# To GUI or Not To GUI?

It is better to learn something new from scratch, without much help from integrated development environments (IDEs), in my experience because that is the quickest method to understand how a certain platform operates.