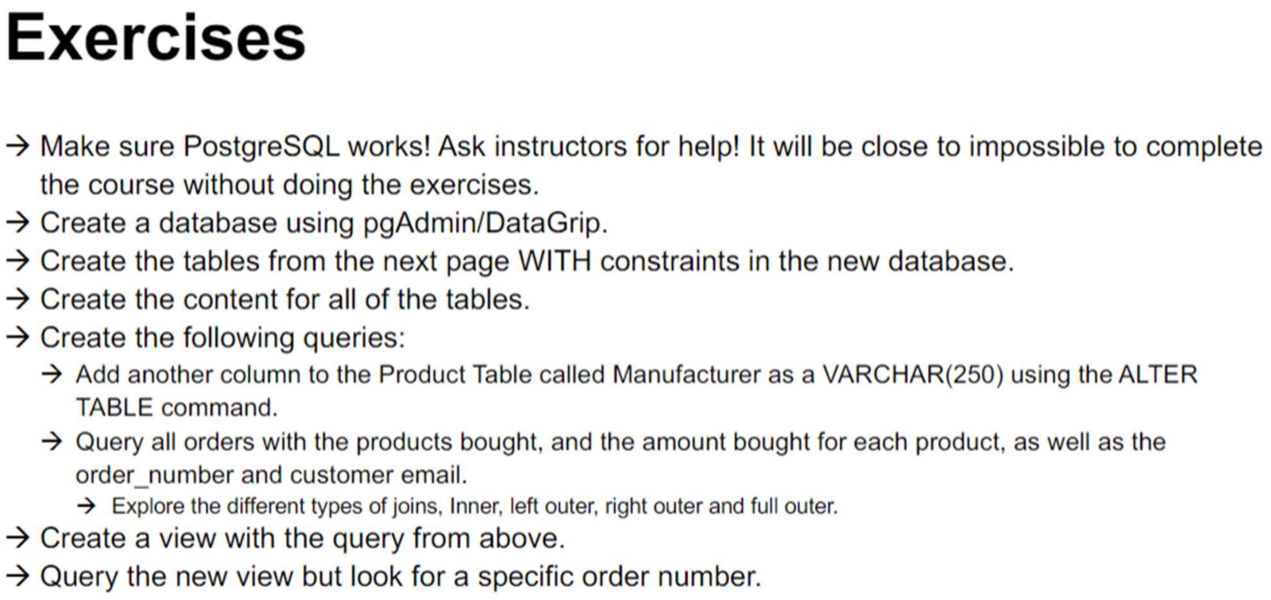
Lektionsøvelse 2



Exercises

→ Make sure PostgreSQL works! Ask instructors for help! It will be close to impossible to complete the course without doing the exercises.
→ Create a database using pgAdmin/DataGrip.
→ Create the tables from the next page WITH constraints in the new database.
→ Create the content for all of the tables.
→ Create the following queries:
  → Add another column to the Product Table called Manufacturer as a VARCHAR(250) using the ALTER TABLE command.
  → Query all orders with the products bought, and the amount bought for each product, as well as the order_number and customer email.
    → Explore the different types of joins, Inner, left outer, right outer and full outer.
→ Create a view with the query from above.
→ Query the new view but look for a specific order number.

Opgave 1

Make sure PostgreSQL work! Ask Instructors for help! It will be close to impossible to complete the course without doing the exercises.

Besvarelse

This part has been done in the first Task, where we had published our code from DataGrip.

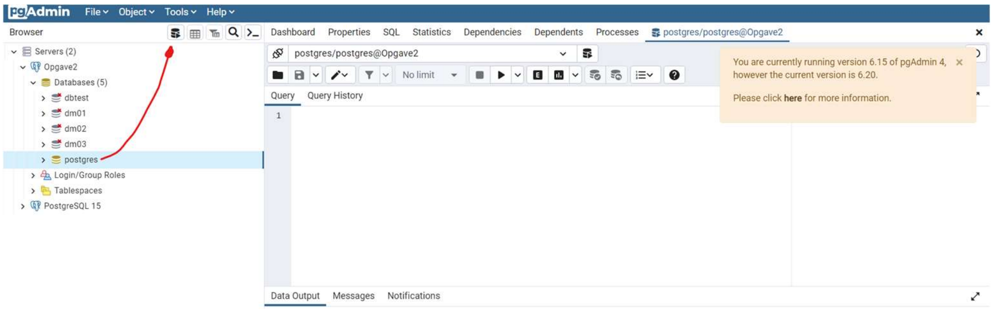
Please remember, that whenever you create a new data source first write postgres on the username and then your password after that. Remember also to test the connection and switch to the latest version of DataGrip.

Opgave 2

Create a database using pgAdmin/Datagrip.

Besvarelse

So, the first thing, we need to do is that we need to open up pgAdmin 4 and then we need to connect it to a localhost server. In this case we have connected it to localhost, where the username is postgres and the localhost is 5432. Down below, we have given a clear view of how to create a database in pgAdmin.



You are currently running version 6.15 of pgAdmin 4, however the current version is 6.20.

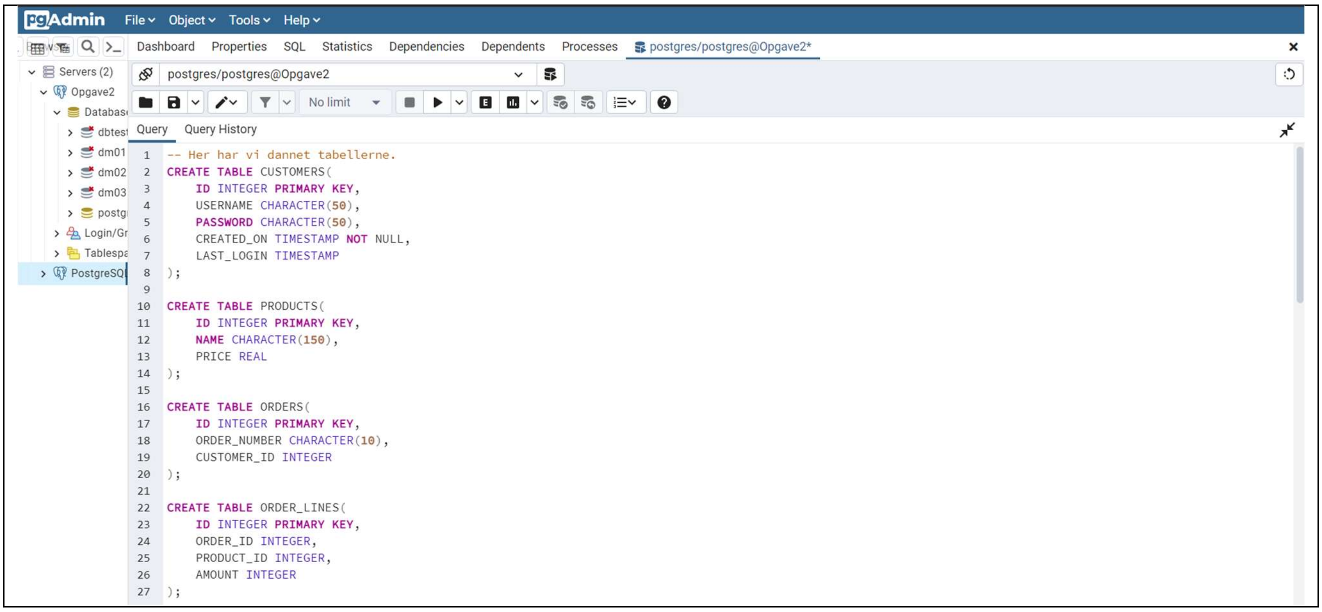
Please click **here** for more information.

Opgave 3

Create the table from the next page with constraints in the new database.

Besvarelse

In this case, we have the following code in which we have created the tables on:



```
-- Her har vi dannet tabellerne.
CREATE TABLE CUSTOMERS(
    ID INTEGER PRIMARY KEY,
    USERNAME CHARACTER(50),
    PASSWORD CHARACTER(50),
    CREATED_ON TIMESTAMP NOT NULL,
    LAST_LOGIN TIMESTAMP
);

CREATE TABLE PRODUCTS(
    ID INTEGER PRIMARY KEY,
    NAME CHARACTER(150),
    PRICE REAL
);

CREATE TABLE ORDERS(
    ID INTEGER PRIMARY KEY,
    ORDER_NUMBER CHARACTER(10),
    CUSTOMER_ID INTEGER
);

CREATE TABLE ORDER_LINES(
    ID INTEGER PRIMARY KEY,
    ORDER_ID INTEGER,
    PRODUCT_ID INTEGER,
    AMOUNT INTEGER
);
```

Opgave 4

Create the content for all queries.

Besvarelse

In this case, we have added all the values for the tables, which are shown above. It is important to note, that we have not added values for id's and times as their special constraints and do require any insertion of a value.

```
-- Nu indsætter vi værdierne ind i Customers.
INSERT INTO CUSTOMERS(USERNAME, PASSWORD,CREATED_ON,LAST_LOGIN)
VALUES ('John','myPassW0rd','john@acme.com',NOW());

INSERT INTO CUSTOMER(USERNAME, PASSWORD, CREATED_ON, LAST_LOGIN)
VALUES ('Anne','SomePassword','anne@acme.com',NOW());

-- Nu indsætter vi værdierne ind i Products.
INSERT INTO PRODUCTS(NAME, PRICE)
VALUES ('Samsung Galaxy S20',7799.95);

INSERT INTO PRODUCTS(NAME, PRICE)
VALUES ('Samsung Galaxy S20 - Leather',799.95);

INSERT INTO PRODUCTS(NAME, PRICE)
VALUES ('IPhone 11 Pro',8899);

INSERT INTO PRODUCTS(NAME, PRICE)
VALUES ('IPhone 11 Pro - Leather Cover',399.5);
```

```
INSERT INTO PRODUCTS(NAME, PRICE)
VALUES ('Huawai P30 Lite',1664.5);

INSERT INTO PRODUCTS(NAME, PRICE)
VALUES ('Huawai P30 - Leather Cover', 1664.5);

-- Nu indsætter vi værdierne ind i Orders.
INSERT INTO ORDERS(ORDER_NUMBER, CUSTOMER_ID)
VALUES ('DA-0001234',1);

INSERT INTO ORDERS(ORDER_NUMBER, CUSTOMER_ID)
VALUES ('DA-0001235',1);

INSERT INTO ORDERS(ORDER_NUMBER, CUSTOMER_ID)
VALUES ('DE-0001236',2);

INSERT INTO ORDERS(ORDER_NUMBER, CUSTOMER_ID)
VALUES ('DE-0001237',2);

-- Nu indsætter vi værdierne ind i Order_Lines.
INSERT INTO ORDER_LINES(ORDER_ID, PRODUCT_ID, AMOUNT)
VALUES (1, 1, 2);

INSERT INTO ORDER_LINES(ORDER_ID, PRODUCT_ID, AMOUNT)
VALUES (1, 2, 2);

INSERT INTO ORDER_LINES(ORDER_ID, PRODUCT_ID, AMOUNT)
VALUES (1, 5, 1);

INSERT INTO ORDER_LINES(ORDER_ID, PRODUCT_ID, AMOUNT)
VALUES (3, 3, 2);

INSERT INTO ORDER_LINES(ORDER_ID, PRODUCT_ID, AMOUNT)
VALUES (3, 4, 1);

INSERT INTO ORDER_LINES(ORDER_ID, PRODUCT_ID, AMOUNT)
VALUES (4, 1, 1);
```

Opgave 5

Create the following queries.

Opgave 5.a

Add another column to the Product Table called Manufacturer as a VARCHAR(250) using the ALTER TABLE command

Besvarelse

Here we have created an simple query in which we have modified the table, and added a column to the product table with the name of Manufacturer with the datatype of VARCHAR(250).

```
ALTER PRODUCTS ADD COLLUMN MANUFACTURER VARCHAR(250);
```

Opgave 5.b

Query all orders with the products bought, and the amount bought for each product, as well as the order_number and customer email.

Besvarelse

```
-- Query all orders
SELECT
       customers.id as customerid,
       customers.username as username,
       orders.order_number as orderNr,
       ol.amount as count,
       products.name as productName
    FROM order_lines ol
    INNER JOIN products ON ol.product_id = products.id
    INNER JOIN orders ON ol.order_id = orders.id
    INNER JOIN customers on orders.customer_id = customers.id;

-- Same as above
SELECT customers.id as customerid,
       customers.username as username,
       orders.order_number as orderNr,
       ol.amount as count,
       products.name as productName
FROM orders, order_lines ol, products, customers
WHERE orders.id = ol.order_id
  AND orders.customer_id = customers.id
  AND ol.product_id = products.id;
```

Opgave 5.c

Explore the different types of joins, Inner, left outer, right outer and full outer.

Besvarelse

```
-- Exploring different types of join
-- Left outer join
SELECT * FROM orders
    LEFT JOIN order_lines ON orders.id = order_lines.order_id
    LEFT JOIN customers ON orders.customer_id = customers.id
    LEFT JOIN products ON order_lines.product_id = products.id;

SELECT customers.username, orders.order_number, products.name,
order_lines.amount FROM orders
    LEFT JOIN order_lines ON orders.id = order_lines.order_id
    LEFT JOIN customers ON orders.customer_id = customers.id
    LEFT JOIN products ON order_lines.product_id = products.id;

-- Right outer join
SELECT customers.username,orders.order_number, products.name,
order_lines.amount FROM orders
    RIGHT JOIN order_lines ON orders.id = order_lines.order_id
    RIGHT JOIN customers ON orders.customer_id = customers.id
    RIGHT JOIN products ON order_lines.product_id = products.id;

-- Full outer join
SELECT customers.username, orders.order_number, products.name,
order_lines.amount FROM orders
    FULL JOIN order_lines ON orders.id = order_lines.order_id
    FULL JOIN customers ON orders.customer_id = customers.id
    FULL JOIN products ON order_lines.product_id = products.id;
```