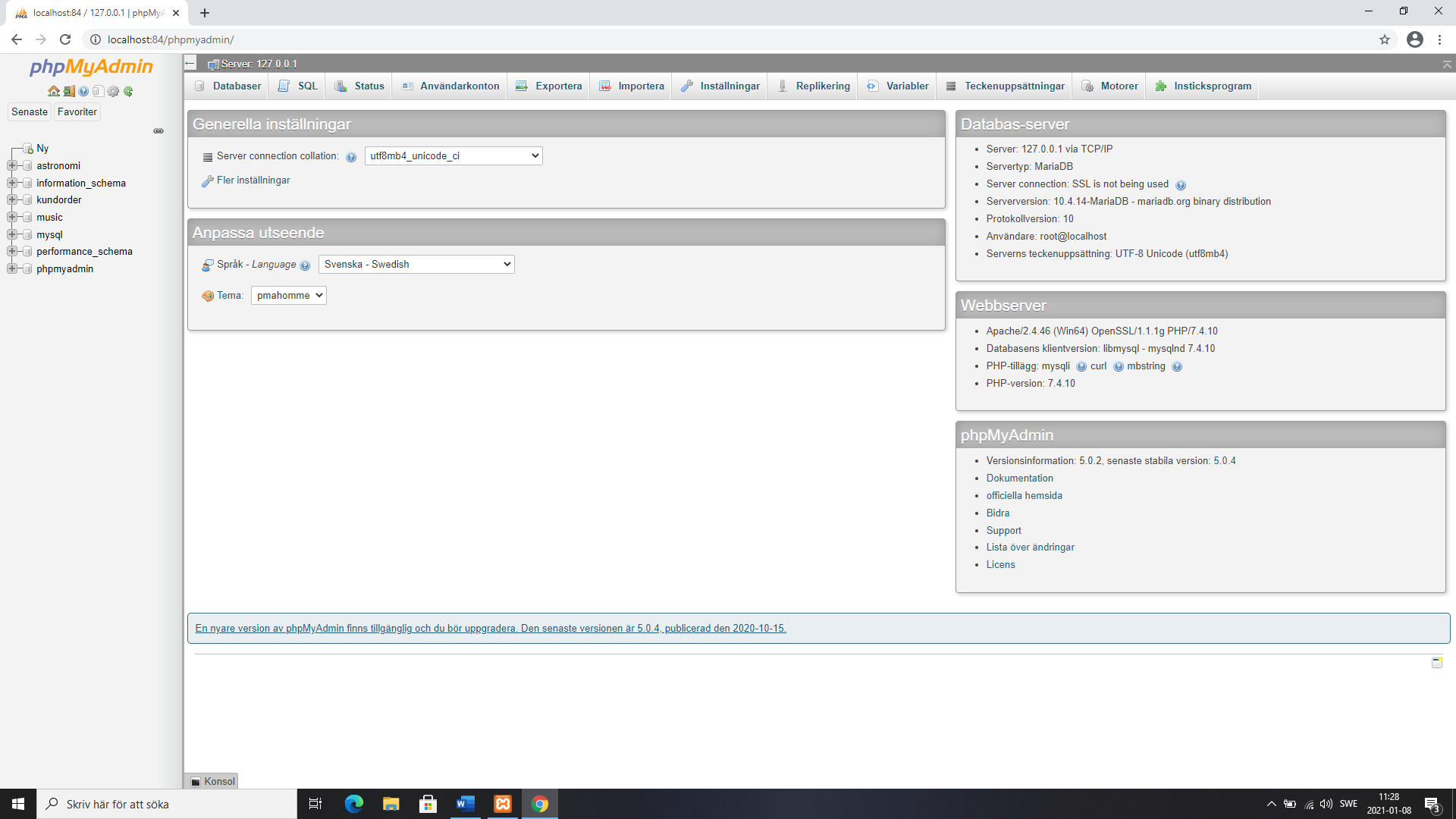
1. Installera xampp
2. Bocka för **starta med kontrollpanel** på sista rutan.
3. Klicka på knappen **Config** längst upp till höger.
4. Autostart: **Apache, MySQL**
5. Klicka på knappen **Service and port settings**.
6. Ändra **Main port** till 84
7. Alternativt: Klicka på knappen **Config** på **Apache** och välj **Apache (httpd.conf)**.
8. Starta Webläsare
9. Skriv in localhost:84/phpmyadmin i adressfältet.

Nedanstående sida bör visas



* Skapa ny databas
* Importera tabeller
  + CSV, utf-8, kolumnseparator ( ; ), Första kolumnen rubrik
* Fixa nycklar, index och relationer
* Skapa queries mot tabell/tabeller

SELECT \* FROM Ordrar ;

SELECT \* FROM kunder WHERE Land = 'sverige';

SELECT kunder.Företagsnamn, kunder.Land FROM kunder WHERE kunder.Land LIKE 's%';

WHERE Orderdatum >= '1997-01-01' AND Orderdatum <= '1997-12-31' ORDER BY Orderdatum ASC;

SELECT \* FROM kunder WHERE Land<> 'Sverige';

SELECT \* FROM produkter WHERE Pris\_ST < 200 OR Antal i lager > 0;

SELECT \* FROM produkter WHERE produktnr > 9 AND Antal i lager > 0;

SELECT \* FROM kunder WHERE Land IN ('Sverige', 'Spanien', 'Schweiz');

SELECT \* FROM kunder WHERE Land NOT IN ('Sverige', 'Spanien', 'Schweiz');

SELECT \* FROM ordrar WHERE LEFT(Orderdatum,4) LIKE '199\_';

SELECT \* FROM ordrar WHERE LEFT(Orderdatum,4) LIKE '199\_' AND RIGHT(Orderdatum,2) = '01';

SELECT \* FROM kunder WHERE Företagsnamn LIKE '%futterkiste #%%' ESCAPE '#';

SELECT \* FROM produkter WHERE Produktnamn REGEXP 'sill';

*Produktnamn som börjar med a, b, c eller d*

SELECT \* FROM Produkter WHERE Produktnamn REGEXP '^[abcd]';

SELECT CONCAT(Företagsnamn, ' (', Land, ')') AS 'Kund och land' FROM kunder;

SELECT Land, COUNT(Företagsnamn) AS 'Antal FTG per Land' FROM kunder GROUP BY Land;

SELECT Företagsnamn, UCASE(Land) FROM kunder;

SELECT produkter.ProduktNamn, produkter.Pris\_ST, produkter. Antal\_i\_lager, Sum(orderrader.Antal)

FROM produkter INNER JOIN orderrader ON produkter.Produktnr = orderrader.Produktnr

WHERE orderrader.Antal > produkter. Antal\_i\_lager

GROUP BY produkter.ProduktNamn, produkter.Pris\_ST, produkter.Antal\_i\_lager;

SELECT kunder.Företagsnamn, kunder.Land, ordrar.Ordernr, ordrar.Orderdatum

FROM Kunder INNER JOIN ordrar ON kunder.Kundnr = ordrar.Kundnr

WHERE LEFT(Orderdatum,4) REGEXP '1997' AND Land Like 's%'

ORDER BY Land, Orderdatum;

SELECT produkter.Produktnamn, produkter.Pris\_ST, produkter.Antal\_i\_lager, Sum(orderrader.Antal)

AS 'Sum Best antal', produkter.Antal\_i\_lager - Sum(orderrader.Antal) AS 'Nytt lagersaldo',

IF(produkter.Antal\_i\_lager - Sum(orderrader.Antal) < 0, 'Restorder', 'Lagersaldo ok') AS 'Koll lagersaldo'

FROM produkter INNER JOIN orderrader ON produkter.Produktnr = orderrader.Produktnr

GROUP BY produkter.Produktnamn;

SELECT kunder.Företagsnamn, ordrar.Ordernr, orderrader.Antal, produkter.ProduktNamn, produkter.Pris\_ST, orderrader.Antal \* produkter.Pris\_ST As 'Radsumma'

FROM kunder

INNER JOIN ordrar ON ordrar.Kundnr = kunder.Kundnr

INNER JOIN orderrader ON orderrader.Ordernr = ordrar.Ordernr

INNER JOIN produkter ON orderrader.Produktnr = produkter.Produktnr;

SELECT kunder.Företagsnamn, kunder.Ort, kunder.Land,ordrar.Ordernr

FROM kunder

LEFT JOIN ordrar ON ordrar.Kundnr = kunder.Kundnr

WHERE ordrar.Ordernr IS NULL

SELECT 23 DIV 6 ; ( 3 )

SELECT 23 / 6 ; ( 3.8333 )

SELECT 23 MOD 6 ; ( 5 )

SELECT FLOOR(23 / 6) AS floor\_result; ( 3 )

SELECT ROUND(23 / 6) AS round\_result; ( 4 )

SELECT RAND() AS random\_result; (decimaltal mellan 0 och 1)

SELECT FLOOR(RAND() \* 10 + 1) AS random\_result;

*Egen definierad funktion*

DELIMITER |

CREATE FUNCTION sf\_DatumKoll(Indata DATE)

RETURNS VARCHAR(15)

BEGIN

DECLARE sf\_value VARCHAR(15);

IF Year(Indata) = 1998 THEN

SET sf\_value = 'Aktuell';

ELSE

SET sf\_value = 'Arkivera order';

END IF;

RETURN sf\_value;

END |

*Använd funktionen*

SELECT ordrar.Ordernr, ordrar.Orderdatum, ordrar.Leveransland, sf\_DatumKoll(ordrar.Orderdatum)

FROM ordrar

ORDER BY ordrar.Leveransland

*Egen definierad procedur*

DELIMITER |

CREATE PROCEDURE MOMS()

BEGIN

DECLARE Variabel REAL;

SET variabel = 0.25;

SELECT produkter.Produktnamn, ROUND(produkter.Pris\_ST \* Variabel, 2) FROM produkter;

END |

*Använd proceduren*

CALL MOMS();

*Redigeringsfrågor:*

CREATE TABLE Order\_arkiv (Ordernr int(11),Orderdatum varchar(10) DEFAULT NULL, Kundnr varchar(5) DEFAULT NULL, AnstNr int(1) , LevOrt varchar(15) DEFAULT NULL, LevLand varchar(14) DEFAULT NULL, Status varchar(14) DEFAULT NULL, PRIMARY KEY (ordernr))

INSERT INTO Order\_arkiv (Ordernr ,Orderdatum, Kundnr, AnstNr, LevOrt, LevLand)

SELECT Ordernr ,Orderdatum, Kundnr, Anställningsnr, Leveransort, Leveransland

FROM ordrar

WHERE ordrar.Orderdatum < '1997-01-01';

DELETE FROM Order\_arkiv WHERE LevLand = 'USA' OR LevLand = 'Brasilien';

UPDATE Order\_arkiv SET LevLand = UCASE (LEFT (LevLand, 3));

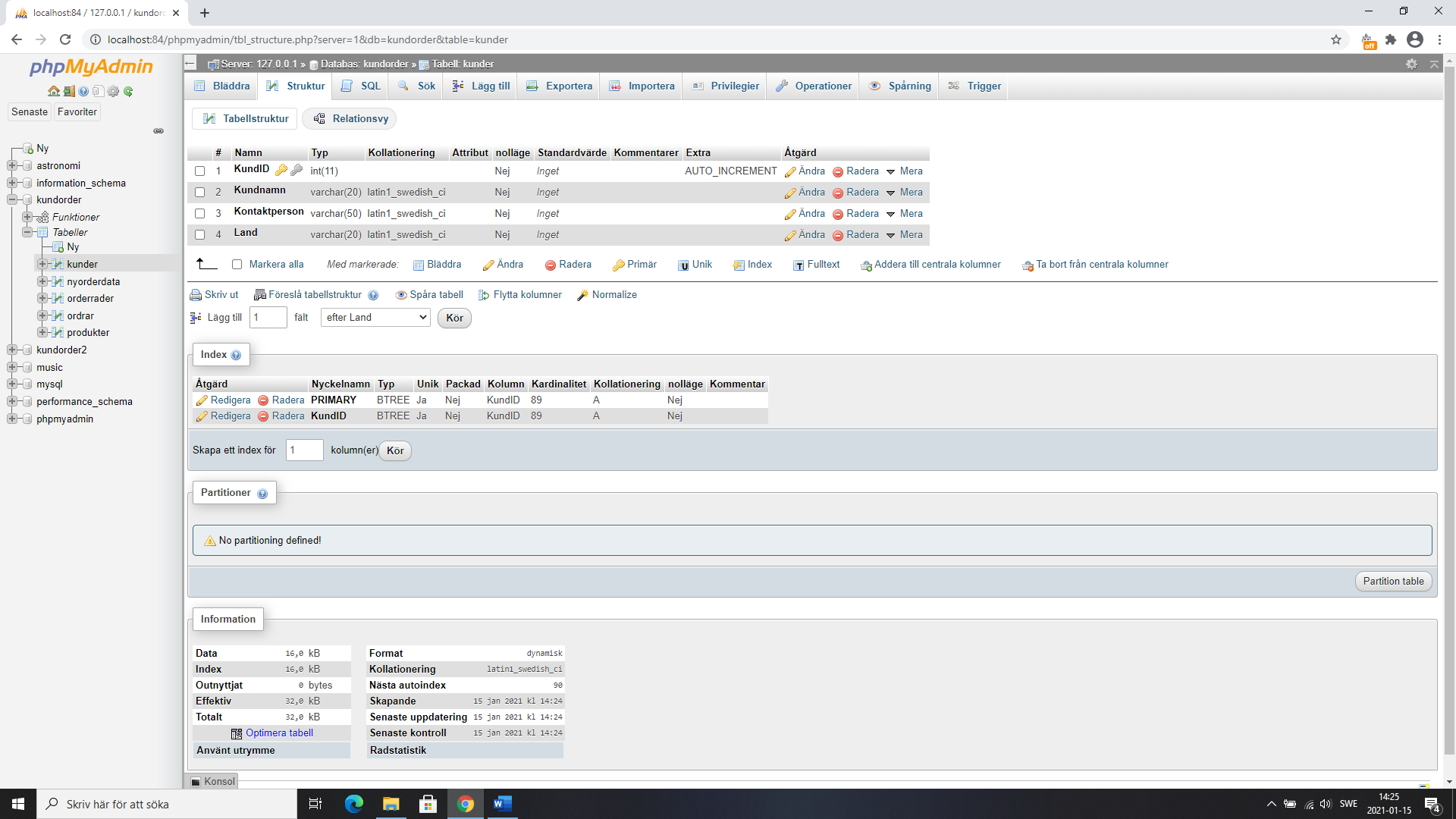
UPDATE Order\_arkiv SET Status = 'Makulerad', LevLand = 'Unknown', LevOrt = 'Unknown'

WHERE Ordernr IN (10248, 10258, 10268, 10278);

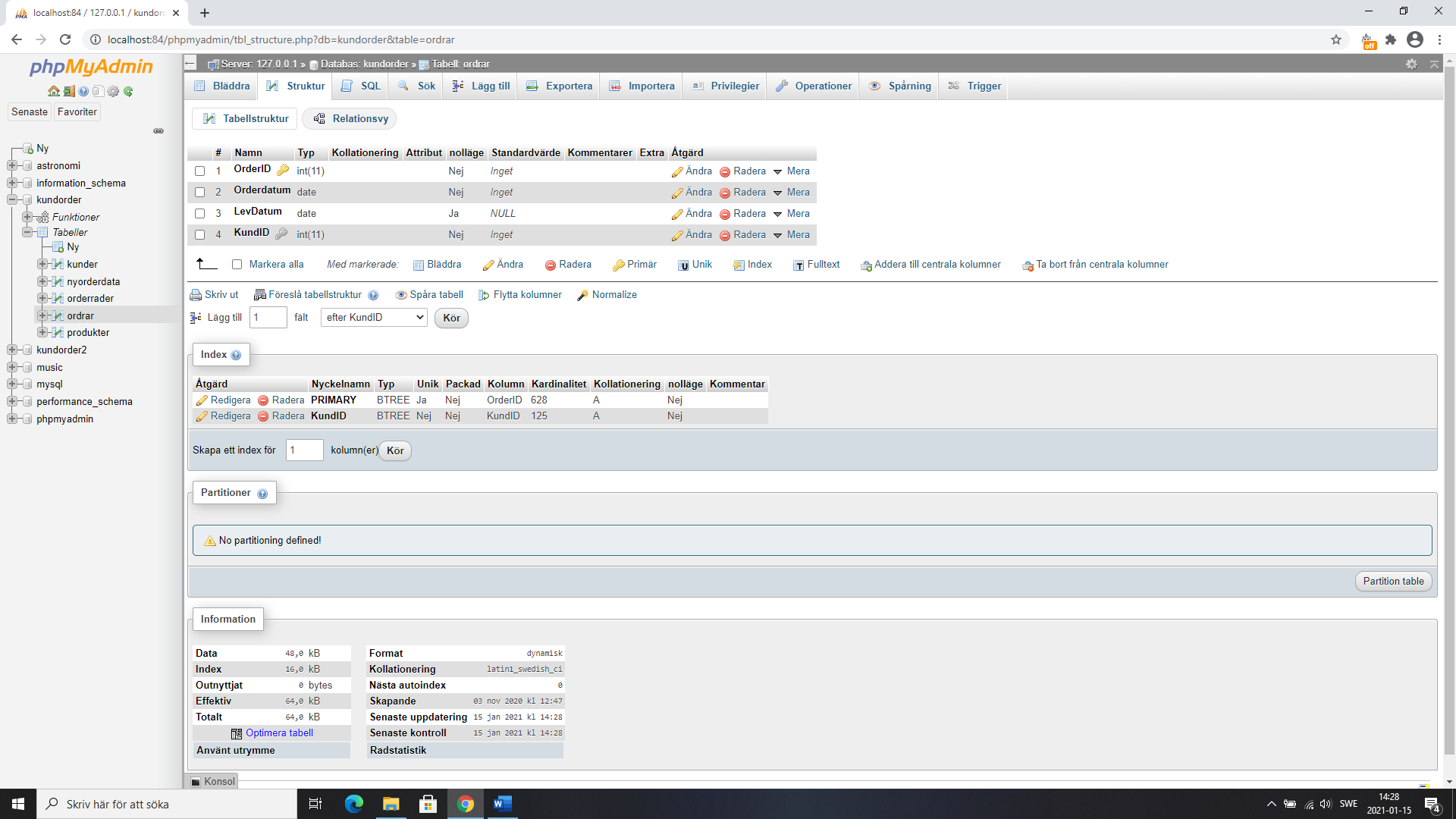
DROP TABLE Order\_arkiv;

*Bygga tabeller med nycklar, index och relationer för import*

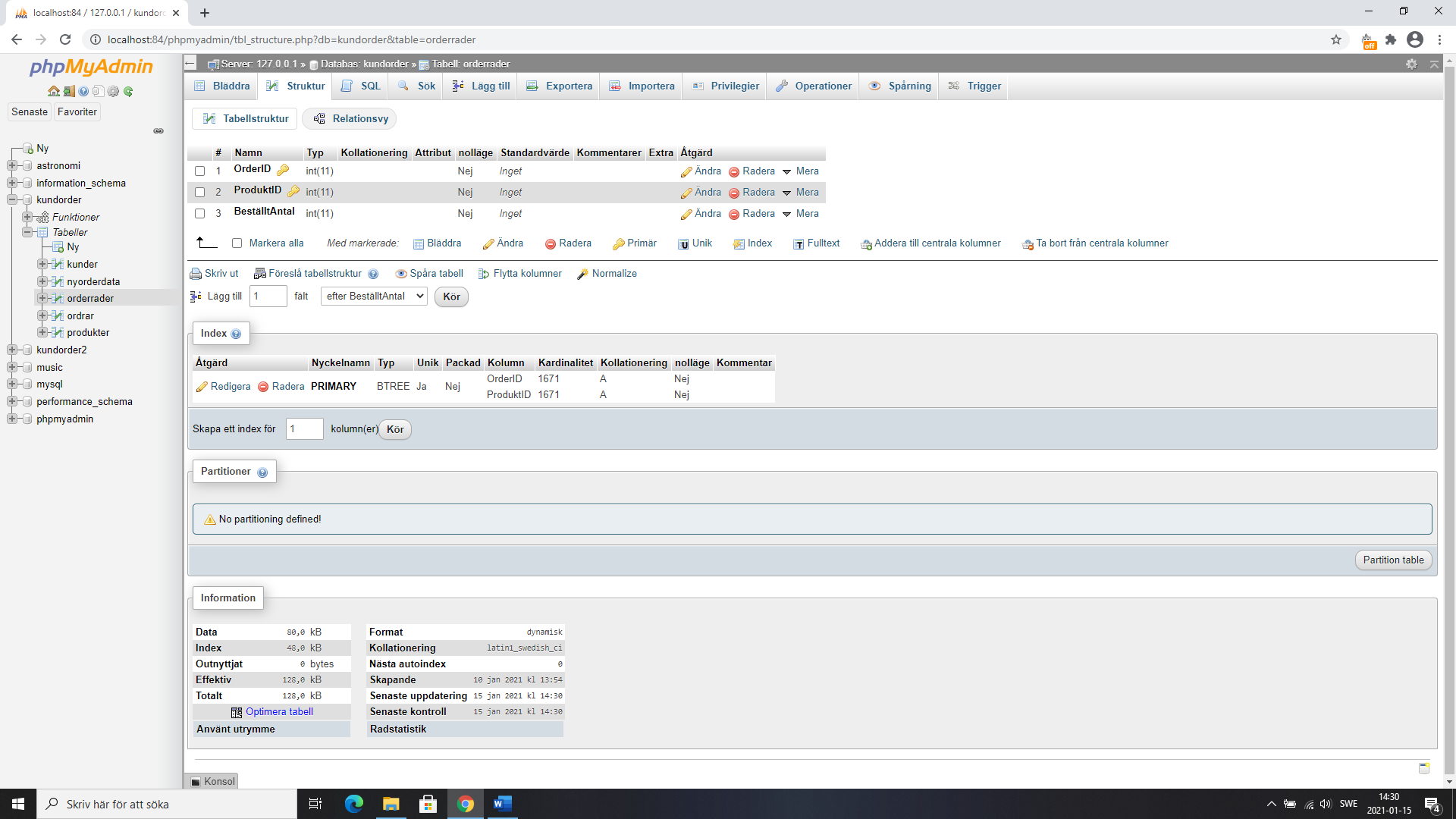
kunder



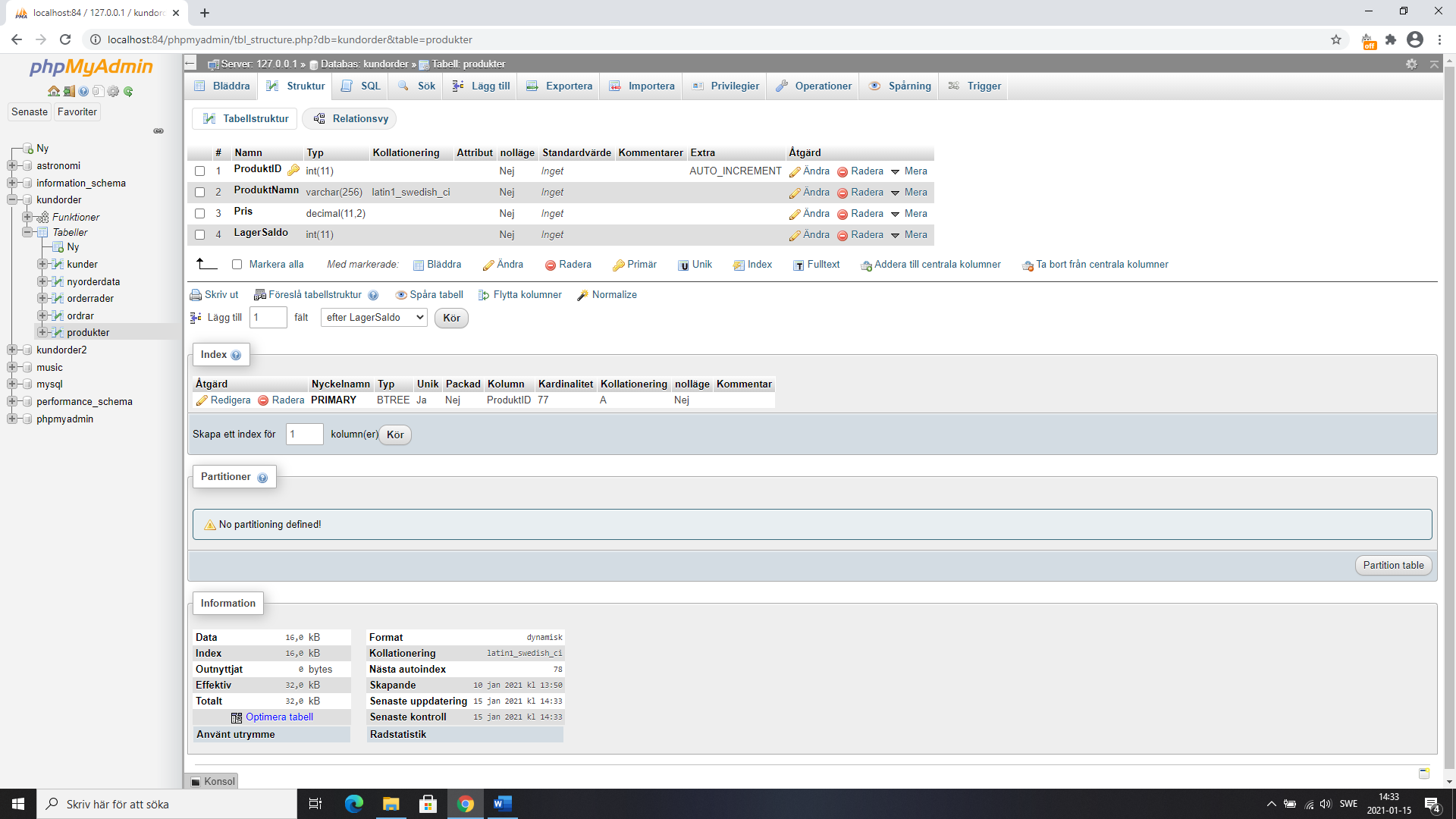
ordrar



orderrader



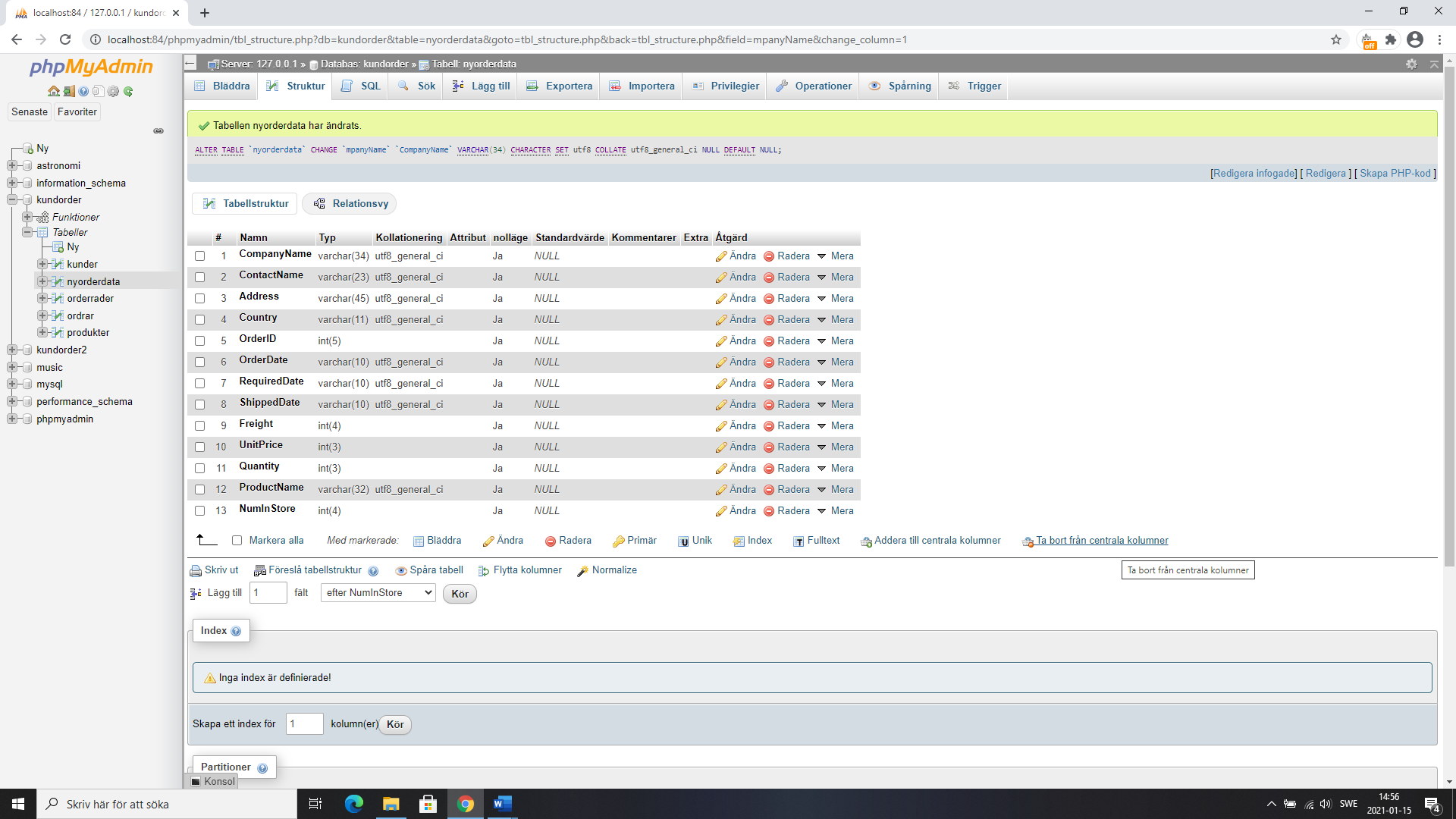
produkter



*Skriv in några poster i tabellerna*

* Tabellen i **Bläddra**-läge
* Klicka på fliken **Lägg till**

*Importera rådata*



*Skapa INSERT INTO Queries med sekundärnycklar*

INSERT INTO kunder ( kunder.Kundnamn, kunder.Kontaktperson, kunder.Land )

SELECT DISTINCT nyorderdata.CompanyName,nyorderdata.ContactName, nyorderdata.Country

FROM NyOrderData

WHERE NOT (CompanyName IN (SELECT kunder.Kundnamn FROM kunder));

INSERT INTO produkter( produkter.ProduktNamn,produkter.Pris,produkter.LagerSaldo)

SELECT DISTINCT NyOrderData.ProductName, nyorderdata.UnitPrice, nyorderdata.NumInStore

FROM NyOrderData

WHERE NOT (ProductName IN (SELECT produkter.ProduktNamn FROM produkter));

INSERT INTO ordrar ( ordrar.OrderID, ordrar.Orderdatum, ordrar.LevDatum,ordrar.KundID)

SELECT NyOrderData.OrderID, NyOrderData.OrderDate, NyOrderData.RequiredDate, kunder.KundID

FROM kunder INNER JOIN NyOrderData ON Kunder.Kundnamn = NyOrderData.CompanyName

GROUP BY NyOrderData.OrderID, kunder.KundID, NyOrderData.OrderDate, NyOrderData.RequiredDate

ORDER BY NyOrderData.OrderID;

INSERT INTO orderrader ( BeställtAntal, ProduktID, OrderID )

SELECT NyOrderData.Quantity, produkter.ProduktID, ordrar.OrderID

FROM ordrar INNER JOIN (produkter INNER JOIN NyOrderData ON produkter.ProduktNamn = NyOrderData.ProductName) ON ordrar.OrderID = NyOrderData.OrderID;

DELIMITER |

CREATE PROCEDURE InfogaData()

BEGIN

INSERT INTO kunder ( kunder.Kundnamn, kunder.Kontaktperson, kunder.Land )

SELECT DISTINCT nyorderdata.CompanyName,nyorderdata.ContactName, nyorderdata.Country

FROM NyOrderData

WHERE NOT (CompanyName IN (SELECT kunder.Kundnamn FROM kunder));

INSERT INTO produkter( produkter.ProduktNamn,produkter.Pris,produkter.LagerSaldo)

SELECT DISTINCT NyOrderData.ProductName, nyorderdata.UnitPrice, nyorderdata.NumInStore

FROM NyOrderData

WHERE NOT (ProductName IN (SELECT produkter.ProduktNamn FROM produkter));

INSERT INTO ordrar ( ordrar.OrderID, ordrar.Orderdatum, ordrar.LevDatum,ordrar.KundID)

SELECT NyOrderData.OrderID, NyOrderData.OrderDate, NyOrderData.RequiredDate,

kunder.KundID

FROM kunder INNER JOIN NyOrderData ON Kunder.Kundnamn = NyOrderData.CompanyName

GROUP BY NyOrderData.OrderID, kunder.KundID, NyOrderData.OrderDate,

NyOrderData.RequiredDate

ORDER BY NyOrderData.OrderID;

INSERT INTO orderrader ( BeställtAntal, ProduktID, OrderID )

SELECT NyOrderData.Quantity, produkter.ProduktID, ordrar.OrderID

FROM ordrar INNER JOIN (produkter INNER JOIN NyOrderData ON produkter.ProduktNamn =

NyOrderData.ProductName) ON ordrar.OrderID = NyOrderData.OrderID;

END |

*Kolla dubletter:*

SELECT kunder.Kundnamn, COUNT(kunder.Kundnamn)

FROM kunder

GROUP BY kunder.Kundnamn

HAVING Count(kunder.Kundnamn) > 1

ORDER BY kunder.Kundnamn;

DELIMITER |

CREATE FUNCTION sp\_Calc\_frakt(ordervarde FLOAT)

BEGIN

DECLARE Frakt FLOAT;

IF ordervarde > 2000 THEN

SET Frakt = 1.05;

ELSEIF ordervarde <= 2000 THEN

SET Frakt = 1.1;

END IF;

RETURN Frakt \* ordervarde;

END |

DELIMITER |

CREATE PROCEDURE sp\_Calc\_frakt()

BEGIN

SELECT kunder.Kundnamn, ordrar.OrderID, ordrar.Orderdatum, ROUND(sf\_Frakt(SUM(orderrader.BeställtAntal \* produkter.Pris)),2) as 'Ordervärde ink frakt', ROUND(SUM(orderrader.BeställtAntal \* produkter.Pris),2) AS 'Ordervärde exkl frakt'

From kunder INNER JOIN ordrar ON kunder.KundID = ordrar.KundID

INNER JOIN orderrader ON ordrar.OrderID = orderrader.OrderID

INNER JOIN produkter ON orderrader.ProduktID = produkter.ProduktID

GROUP BY kunder.Kundnamn, ordrar.OrderID

ORDER BY kunder.Kundnamn, ordrar.OrderID;

END |

Uppgift 1 MySql

Företaget **XTreme** håller på med cykelsport. De skall migrera sin data från en SQL Server till MySql eftersom företaget så småningom skall bygga en e-handelsplattform utifrån en Apache-server.

De har exporterat all data till CSV-filer för att enkelt kunna importera datat till MySql. Alla nyckelfält finns i exportfilerna för att enkelt kunna koppla ihop tabellerna med relationer.

*Ert jobb är att:*

1. Städa importfilerna (Inga mellanslag eller konstiga tecken i fältnamn, Ersätt tomma värden i fält med lämpliga värden, Inga mellanslag i tabellnamnen)  
     
   Customer.csv

Orders Detail.csv

Orders.csv

Product Type.csv

Product.csv

Supplier.csv

1. Skapa en ny databas i MySql.
2. Importera CSV-filerna till databasen.
3. Se till att varje tabell har en primärnyckel och att primärnyckelfältet är indexerat.
4. Analysera hur tabellerna skall kopplas ihop och skapa relationer. Tänk på att även indexera sekundärnycklarna.
5. Skapa följande frågor för utdata genom att använda SQL-verktyget.  
     
   1. Visa alla kunder från USA.

2. Visa alla leverantörer från Japan och deras produkter.

3. Visa alla produkter för produkttyperna *Mountain* och *Competition*.

4. Visa totalt ordervärde per kund.

5. Visa antal produkter per produkttyp.

6. Visa totalt ordervärde per order.

Uppgift 2 MySql

Ni har fått i uppgift att hjälpa **NASA** att bygga en databas för att lagra data om astronomiska objekt.

*Följande uppgifter skall finnas med:*

Constellations (Stjärnbilder), Stars, DeepSkyObjects(galaxer, nebulosor, etc), Planets, Moons, Spectral\_class(klassificering av stjärnor i färg och temperatur).

*Följande gäller:*

* En planet kan ha flera månar men bara tillhöra en stjärna.
* En Constellation(stjärnbild) kan innehålla flera Stjärnor och flera DeepSkyObjects(galaxer,nebulosor etc).
* En Stjärna kan bara tillhöra en Spektralklass.

Ni skall använda följande tre importfiler för att populera tabellerna med data.

DeepSkyConstellations.csv, PlanetsMoons.csv, StarsConstellations.csv

*Gör följande:*

1. **Skapa databas**

Skapa en ny databas och döp den till *Astronomy*

1. **Skapa tabeller med primärnycklar**

Följande fält skall finnas med för att kunna ta emot data från importfilerna:

Constellation, DeepSkyObject, MagnitudeDeepSky, Ra\_Dec, Distance\_LY, NGC, Messier, Star, Planet, Diameter, Temperature\_K, Temperature\_C, Moon, MoonDistance, MoonPeriod, MoonMagnitude, Hemisfere(halvklot), Planets\_in\_Const, Star, StarDistance\_LY, Spectral\_Class, Colour, Temperature

*Färgerna på fälten i importfilerna indikerar vad som hör ihop.*

|  |  |  |
| --- | --- | --- |
| Importfil: DeepSkyConstellations.csv | Importfil: PlanetsMoons.csv | Importfil: StarsConstellations.csv |
| Constellation  DeepSkyObject  Magnitude  Ra\_Dec  Distance\_LY  NGC  Messier | Star  Planet  Diameter  Temperature\_K  Temperature\_C  Moon  Distance\_1000km  Period\_(days)  Magnitude | Constellation  Hemisfere  Planets\_in\_Const  Star  Distance\_LY  Spectral\_Class  Colour  Temperature |

Skapa numeriska primärnyckelält som autogenererar heltal. Skapa även sekundärnycklar som kan kopplas till primärnycklarna. Indexera alla nyckelfält.

1. **Skapa relationer mellan tabellerna**

Analysera vilka tabeller som skall kopplas och hur de skall kopplas. Skapa relationer mellan nyckelfälten i tabellerna.

1. **Importera data till tabellerna**

Skriv sql-kod som importerar data från importfilerna. Tänk på att importera data på ”ett-sidan” först, dvs yttertabbellerna först. Importera utifrån och innåt i relationsstrukturen. På detta sätt skapas primärnyckelvärden som kan användas för att populera sekundärnycklarna.

Använd Insert-statements för att göra jobbet.

1. **Skapa följande sql-frågor för utdata**
2. Visa Constellations(stjärnbilder) med deepSky-objekt för norra halvklotet.
3. Visa planeter med månar.
4. Visa planeter utan månar.
5. Visa alla stjärnor som inte tillhör spektralklass W.
6. Visa antal stjärnor per stjärnbild.
7. Visa alla stjärnbilder med funna exoplaneter.
8. **Skapa en stored function**

Skapa en funktion som beräknar restiden i år till stjärnorna. Funktionen skall ha en inparameter för avstånd. Vi antar att hastigheten är 100 000 km/h.

1. **Skapa en stored procedure**

Skapa en procedur med sql-kod som kör funktionen. Sql-koden skall även visa stjärnans namn och avstånd.