1 A) DBMS
Databas ledningssystem är ett mjukvarusystem som möjliggör användaren att skapa, lagra, hämta, läsa, uppdatera och ta bort data. Ett gränssnitt mellan slutanvändare och databas.

B) Redundans
Förekommer när data dubbleras en eller flera gånger och lagras på flera ställen

C) Normalisering tom 3NF
Sätt att utforma scheman till databaser. 1NF: Första normala formen, där är varje kolumn helt unik. 2NF: Andra normala formen, där någon av 1NF:s kolumner är primary key till nya tabellen och de andra kolumnerna rättar sig efter denna. 3NF: När tabellen är en 2NF men kolumnerna är självständiga och inte beroende av primary key.

D) AVG
En funktion som ger tillbaka medelvärdet.

2. Vad är skillnaden mellan SQL och NoSql, dess fördelar och nackdelar.
SQL är en databas där tabeller har en koppling till varandra Det har inte NOSQL. Fördelar med SQL är att det är enkelt att använda, det är universellt, bra för att strukturera data och högpresterande arbeten. Det negativa med SQL är att det kan vara tidskonsumerande att sätta sig in i att förstå och skapa strukturen, kan också vara svårt att väga mot övrig information.
fördelar med NoSQL är att man slipper skapa mallar, ständigt snabba och förbättrade cykler, mindre tidskrävande än SQL samt att det fungerar ihop med cloud. Det negativa med NoSQL är att den inte är kapabel till att sammankoppla data samt att det svarar i låg hastighet.

3. Vad är Json och vad används det till. Ge ett exempel när det är lämpligt att använda XML JSON kommer från JavaScrip och står för Java Script Object Notation. Optimalt vid förflyttning av data från server till webbplats. Exempel på XML: Man mer hjälp av JavaScript läsa en XML-fil och komplettera innehållet på alla HTML-platser.

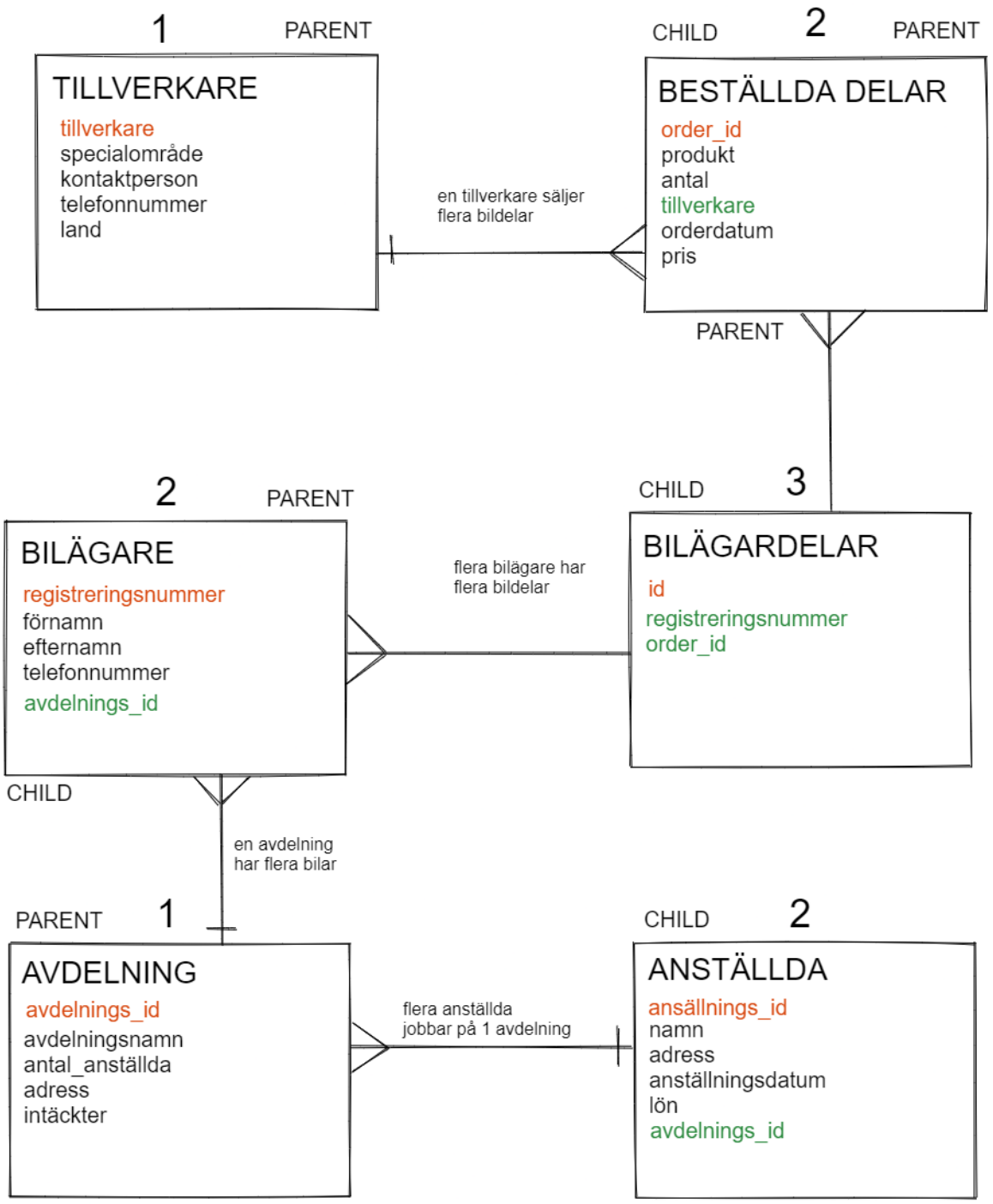
4. Vad är dataintegritet och hur upprätthåller en det? Dataintegritet är att se till att datan är autentisk och stämmer. För att upprätthålla detta bör man ta bort dubbletter, hitta saknad, föråldrad eller felaktig data. Man skall säkerhetskopiera och övervaka utbudet av data samt dokumentera stegen man gör.

5. Beskriv enkelt SQL servers uppbyggnad Det är ett relationellt databashanteringssystem som möjliggör skapandet av databaser med tabeller som är kopplade till varandra.

6. Vad är en tabell och vad är skillnaden mellan en vy? Tabeller är ett objekt där kolumner och rader är logiskt kopplade till varandra och som är sparad och kan hämtas vid behov. En vy, däremot är det resultat som grundas på sökningen från en SQL-sats och som ej sparas.

7. Förklara skillnaden mellan att ha logiken i databasen kontra applikationen Skillnaden är att med logiken i databas betyder att man skapar en datamodell som visar information från tex ett företag. Medan applikation antyder på den programvara som kör databassystemet eller applikationer som är kopplade till en viss databas för att hjälpa slutanvändaren att tillhandahålla delar av databasen.

DIAGRAM



1. "Tillverkare" och "Avdelning" har ingen foreign key, dessa är parents och skapas först.

2. "Anställda" har "Avdelning_id" från tabell "Avdelning" som foreign key. "Beställda delar" har foreign key från "Tillverkare" med kolumnen "Tillverkare" som foreign key och "Bilägare" har "avdelnings_id" från "Avdelning som foreign key. Dessa 3 är child och skapas som nummer två.

3. Eftersom "Bilägare" och "Beställda delar" har en flera till flera-relation skapas en tredje tabell med registreringsnummer från "Bilägare" och "order_id" från "beställda delar" som foreign key. Denna tabell är en child och "Bilägare" och "Beställda delar" blir då parent för denna. Alltså är "bilägare" och "Beställda delar" både child och parent.

```
-- Skapa en databas om den existerar, uppdatera och använd databas
USE [master];

IF EXISTS (SELECT * FROM sys.databases WHERE name = 'inlämning')
BEGIN
  DROP DATABASE inlämning;
END;

CREATE DATABASE inlämning;
GO

USE [inlämning];
GO

-- TILLVERKARE Skapa tabeller och för in värden, denna har ingen foreign key eftersom man ska börja med de som inte har det.(parent-tabell)
CREATE TABLE tillverkare (
        tillverkare VARCHAR (25) PRIMARY KEY,
        specialområde VARCHAR (25) NOT NULL,
        kontaktperson VARCHAR (25) DEFAULT NULL,
        telefonnummer VARCHAR (25) NOT NULL,
        land VARCHAR (25) DEFAULT NULL
);
-- lägger till värden i tabellen
INSERT INTO tillverkare(tillverkare, specialområde, kontaktperson, telefonnummer, land) VALUES ('Scanda Audi motors', 'Audimotorer', NULL, '+438850234', 'Damnark');
INSERT INTO tillverkare(tillverkare, specialområde, kontaktperson, telefonnummer, land) VALUES ('Automobil SLK', 'cylindrar', 'Frans Shulser', '+478937522', 'Tyskland');
INSERT INTO tillverkare(tillverkare, specialområde, kontaktperson, telefonnummer, land) VALUES ('Die decken firmen', 'däck', NULL, '+47233789784', 'Tyskland');
INSERT INTO tillverkare(tillverkare, specialområde, kontaktperson, telefonnummer, land) VALUES ('Norrlands däck', 'däck till Motorla', 'Klas Malm', '028850234', 'Sverige');
INSERT INTO tillverkare(tillverkare, specialområde, kontaktperson, telefonnummer, land) VALUES ('Skrachna automobil', 'polering & Interiör', NULL, '+446493728', 'Polen');


-- AVDELNING. Denna tabell har heller ingen foreign key. Detta är också en parent-tabell
CREATE TABLE avdelning (
        avdelnings_id INT PRIMARY KEY,
        område VARCHAR (25) DEFAULT NULL,
        antal_anställda INT DEFAULT NULL,
        intäkter INT DEFAULT NULL,
        antal_servrade_bilar_2021 INT DEFAULT NULL
);
-- lägger till värden i tabellen
INSERT INTO avdelning(avdelnings_id, område, antal_anställda, intäkter, antal_servrade_bilar_2021) VALUES (1, 'Däckbyte', 18, 8850234, 2950);
```

```
INSERT INTO avdelning(avdelnings_id, område, antal_anställda, intäkter, antal_servrade_bilar_2021) VALUES (2, 'Motorer', 14, 7902700, 2634);
INSERT INTO avdelning(avdelnings_id, område, antal_anställda, intäkter, antal_servrade_bilar_2021) VALUES (3, 'Bakhjulscylindrar', 27, 6819034, 2273);
INSERT INTO avdelning(avdelnings_id, område, antal_anställda, intäkter, antal_servrade_bilar_2021) VALUES (4, 'Lack och polering',31, 5631220, 1877);
INSERT INTO avdelning(avdelnings_id, område, antal_anställda, intäkter, antal_servrade_bilar_2021) VALUES (5, 'Interiör', 16, 4701040, 1562);

-- BESTÄLLDA DELAR. Denna tabell har en foreign key från tabellen tillverkare, en child-tabell, men det är också en parent-tabell till bilägardelar
CREATE TABLE beställda_delar (
        order_id INT NOT NULL PRIMARY KEY,
        produkt VARCHAR (40) DEFAULT NULL,
        antal INT DEFAULT NULL,
        tillverkare VARCHAR (25) NOT NULL,
        orderdatum DATETIME NOT NULL,
        pris INT NOT NULL,
        FOREIGN KEY (tillverkare) REFERENCES tillverkare (tillverkare) ON DELETE CASCADE ON UPDATE CASCADE
);
-- lägger till värden i tabellen
INSERT INTO beställda_delar(order_id, produkt, antal, tillverkare, orderdatum, pris) VALUES (1, '323d däck', 4,'Die decken firmen', '2022-03-01 10:30:30', 7200);
INSERT INTO beställda_delar(order_id, produkt, antal, tillverkare, orderdatum, pris) VALUES (2, '353 motordel', 3,'Scanda Audi motors', '2022-03-01 10:35:30', 6300);
INSERT INTO beställda_delar(order_id, produkt, antal, tillverkare, orderdatum, pris) VALUES (3, '4391L däck', 2,'Die decken firmen', '2022-12-01 15:45:02',4700);
INSERT INTO beställda_delar(order_id, produkt, antal, tillverkare, orderdatum, pris) VALUES (4, 'cylinder 34K', 4,'Automobil SLK', '2022-11-03 11:04:12', 5600);
INSERT INTO beställda_delar(order_id, produkt, antal, tillverkare, orderdatum, pris) VALUES (5, 'framsätesdel höger32', 4,'Skrachna automobil', '2022-09-11 09:16:10', 1800);

-- ANSTÄLLDA. denna tabell har en foerign key från tabellen avdelning, den sammanslagna kolumnen är avdelnings_id. Detta är en child-tabell
CREATE TABLE anställda (
        anställnings_id INT PRIMARY KEY,
        förnamn VARCHAR (40) DEFAULT NULL,
        efternamn VARCHAR (40) DEFAULT NULL,
        anställningsdatum DATE NOT NULL,
        lön INT NOT NULL,
        kön CHAR (1) NOT NULL,
        avdelnings_id INT NOT NULL,
        FOREIGN KEY (avdelnings_id) REFERENCES avdelning (avdelnings_id) ON DELETE CASCADE ON UPDATE CASCADE
);
-- lägger till värden i tabellen
```

```
INSERT INTO anställda(anställnings_id, förnamn, efternamn, anställningsdatum, lön, kön,
avdelnings_id) VALUES (1, 'Karl', 'Persson', '2018-03-01', 538200, 'M', 3);
INSERT INTO anställda(anställnings_id, förnamn, efternamn, anställningsdatum, lön, kön,
avdelnings_id) VALUES (2, 'Jakob', 'Karlsson', '2018-04-23', 750300, 'M', 5);
INSERT INTO anställda(anställnings_id, förnamn, efternamn, anställningsdatum, lön, kön,
avdelnings_id) VALUES (3, 'Lena', 'Fredrisksson', '2013-07-15', 854800, 'K', 3);
INSERT INTO anställda(anställnings_id, förnamn, efternamn, anställningsdatum, lön, kön,
avdelnings_id) VALUES (4, 'Josef', 'Hadarsson', '2022-06-30', 405000, 'M', 2);
INSERT INTO anställda(anställnings_id, förnamn, efternamn, anställningsdatum, lön, kön,
avdelnings_id) VALUES (5, 'Jesper', 'Lund', '2022-05-01', 773000, 'M', 1);

-- BILÄGARE. Denna tabell har en foreign key,från avdelning. Det gör denna tabell till en child-tabell
men det är också en parent-tabell till bilägardelar
CREATE TABLE bilägare (
        registreringsnummer VARCHAR(6) NOT NULL PRIMARY KEY,
        förnamn VARCHAR (40) DEFAULT NULL,
        efternamn VARCHAR (40) DEFAULT NULL,
        telefonnummer VARCHAR (20) DEFAULT NULL,
        avdelnings_id INT NOT NULL,
        FOREIGN KEY (avdelnings_id) REFERENCES avdelning (avdelnings_id) ON DELETE
CASCADE ON UPDATE CASCADE
);
-- lägger till värden i tabellen
INSERT INTO bilägare(registreringsnummer, förnamn, efternamn, telefonnummer, avdelnings_id)
VALUES ('MDK265','Lars', 'Hellbom', 0743883475, 1);
INSERT INTO bilägare(registreringsnummer, förnamn, efternamn, telefonnummer, avdelnings_id)
VALUES ('YHY782','Hanna', 'Svensson', 0733875512, 3);
INSERT INTO bilägare(registreringsnummer, förnamn, efternamn, telefonnummer, avdelnings_id)
VALUES ('OMD153', 'Harald', 'Lindros', 0737621197, 2);
INSERT INTO bilägare(registreringsnummer, förnamn, efternamn, telefonnummer, avdelnings_id)
VALUES ('HEI322', 'Mikael', 'Forslund', 0737542678, 1);
INSERT INTO bilägare(registreringsnummer, förnamn, efternamn, telefonnummer, avdelnings_id)
VALUES ('IKR244', 'Jimmy', 'Brant', 0707073445, 4);

-- Sist sammanfogar jag many-to-many tabellerna bilägare och beställda_delar till denna tabell. Detta är
en childtabell till dessa.
-- Foreign keys är registreringsnummer från bilägare samt order_id från beställda delar.
CREATE TABLE bilägardelar (
        id INT PRIMARY KEY,
        registreringsnummer VARCHAR(6) NOT NULL,
        order_id INT NOT NULL,
        FOREIGN KEY (registreringsnummer) REFERENCES bilägare (registreringsnummer) ON
DELETE CASCADE ON UPDATE CASCADE,
        FOREIGN KEY (order_id) REFERENCES beställda_delar (order_id) ON DELETE CASCADE
ON UPDATE CASCADE
);
```

```
-- lägger till värden i tabellen
INSERT INTO bilägardelar(id, registreringsnummer, order_id) VALUES (1,'HEI322', 1);
INSERT INTO bilägardelar(id, registreringsnummer, order_id) VALUES (2,'MDK265', 2);
INSERT INTO bilägardelar(id, registreringsnummer, order_id) VALUES (3,'HEI322', 3);
INSERT INTO bilägardelar(id, registreringsnummer, order_id) VALUES (4,'OMD153', 1);
INSERT INTO bilägardelar(id, registreringsnummer, order_id) VALUES (5,'YHY782', 5);

-- SE LISTORNA
SELECT * FROM avdelning;
SELECT * FROM anställda;
SELECT * FROM bilägare;
SELECT * FROM tillverkare;
SELECT * FROM beställda_delar;
SELECT * FROM bilägardelar;

-- Visa allt från en kolumn.
SELECT bilägare.förnamn FROM bilägare;

-- Visar alla på avdelning 3 samt alla som har förnamn 'Jakob'.
SELECT * FROM anställda WHERE avdelnings_id = 3 OR förnamn = 'Jakob';

-- Anställda som fick jobb i mars, datumsökning
SELECT * FROM anställda
WHERE anställningsdatum LIKE '____-03%';

-- Sök efter alla NULL i en specifik tabell
SELECT * FROM tillverkare WHERE kontaktperson IS NULL;

-- Man använder BETWEEN för att visa det som finns mellan 2 besämda värden.
SELECT * FROM avdelning
WHERE antal_servrade_bilar_2021
BETWEEN '2000' AND '3000';

-- Aggregate-funktioner används för räkning, här är den totala summan av alla löner
SELECT SUM (lön) AS Total_lön FROM anställda;

-- Medelvärdet av intäckter för de olika avdelningarna
SELECT AVG (intäkter) FROM avdelning;

-- Ordna efter lön
SELECT * FROM anställda
ORDER BY lön DESC;

-- TOP väljer att visa (i detta fall) 2 rader.
SELECT TOP 2 *
FROM anställda;
```

```
--Räkna ihop allas löner för de anställda
SELECT förnamn, efternamn, lön,
SUM(lön) OVER (ORDER BY lön RANGE BETWEEN UNBOUNDED PRECEDING AND
CURRENT ROW) AS total_lön
FROM anställda;

-- Hur många män respektive kvinnor finns det
SELECT COUNT (kön), kön FROM anställda
GROUP BY kön;

--INNER JOIN Får med allt som stämmer överrens med de olika tabellerna
SELECT * FROM avdelning INNER JOIN anställda
ON avdelning.avdelnings_id=anställda.avdelnings_id

--LEFT JOIN
SELECT * FROM avdelning LEFT JOIN anställda
ON avdelning.avdelnings_id=anställda.avdelnings_id

--Skapa en variabel och fyll på med ett värde
DECLARE @regg_nr VARCHAR(6);
SET @regg_nr='IKR244';

-- Sök med en variabel, först skapar man en variabel med det värdet jag söker, sen väljer man var man vill
söka
DECLARE @regg_nr VARCHAR(6);
SET @regg_nr='IKR244';
SELECT * FROM bilägare
WHERE registreringsnummer = @regg_nr;

-- Lägg till en kolumn i en tabell
ALTER TABLE bilägare
ADD adress VARCHAR(50);

-- Ta bort en rad
Delete FROM bilägare WHERE registreringsnummer = 'IKR244';

-- Byt namn på en tabell
EXEC sp_rename 'bilägare', 'Kund';
```