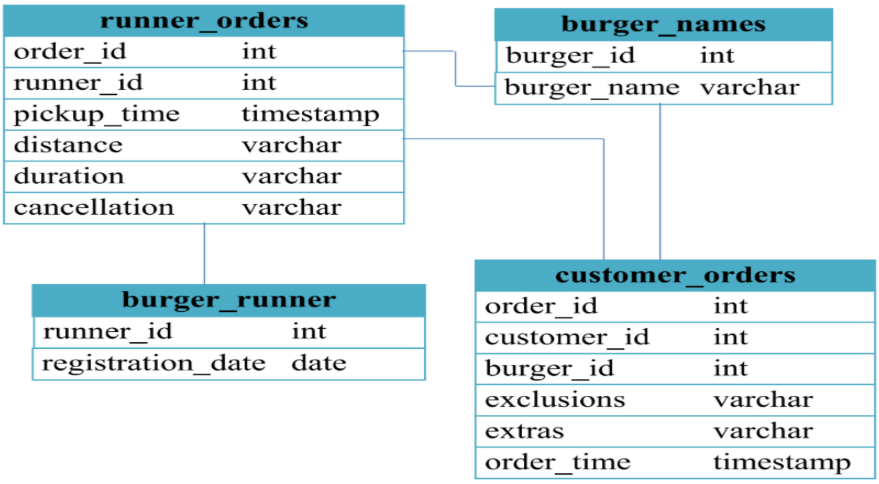
SQL Case Study 2: Burger Bash

SCHEMA USED



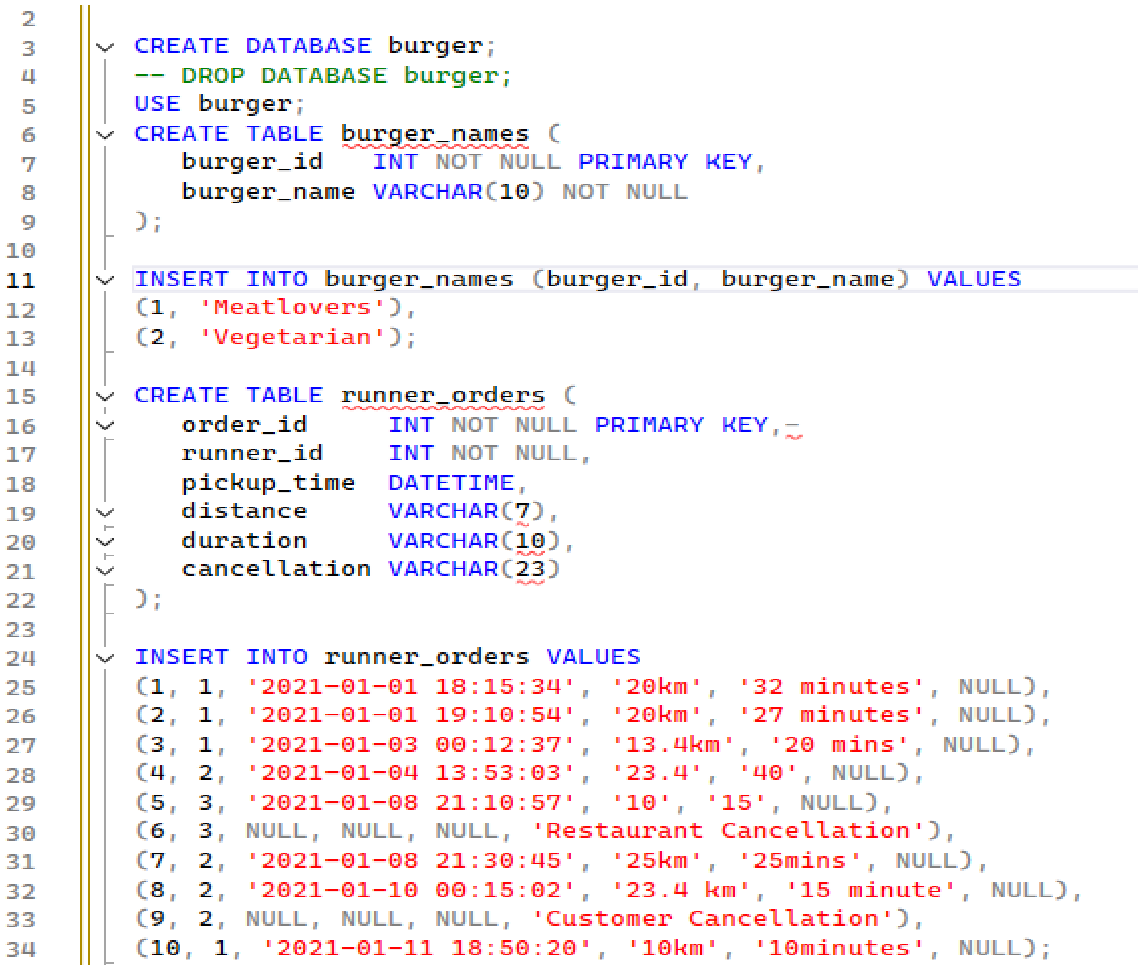
runner_orders	
order_id	int
runner_id	int
pickup_time	timestamp
distance	varchar
duration	varchar
cancellation	varchar

burger_names	
burger_id	int
burger_name	varchar

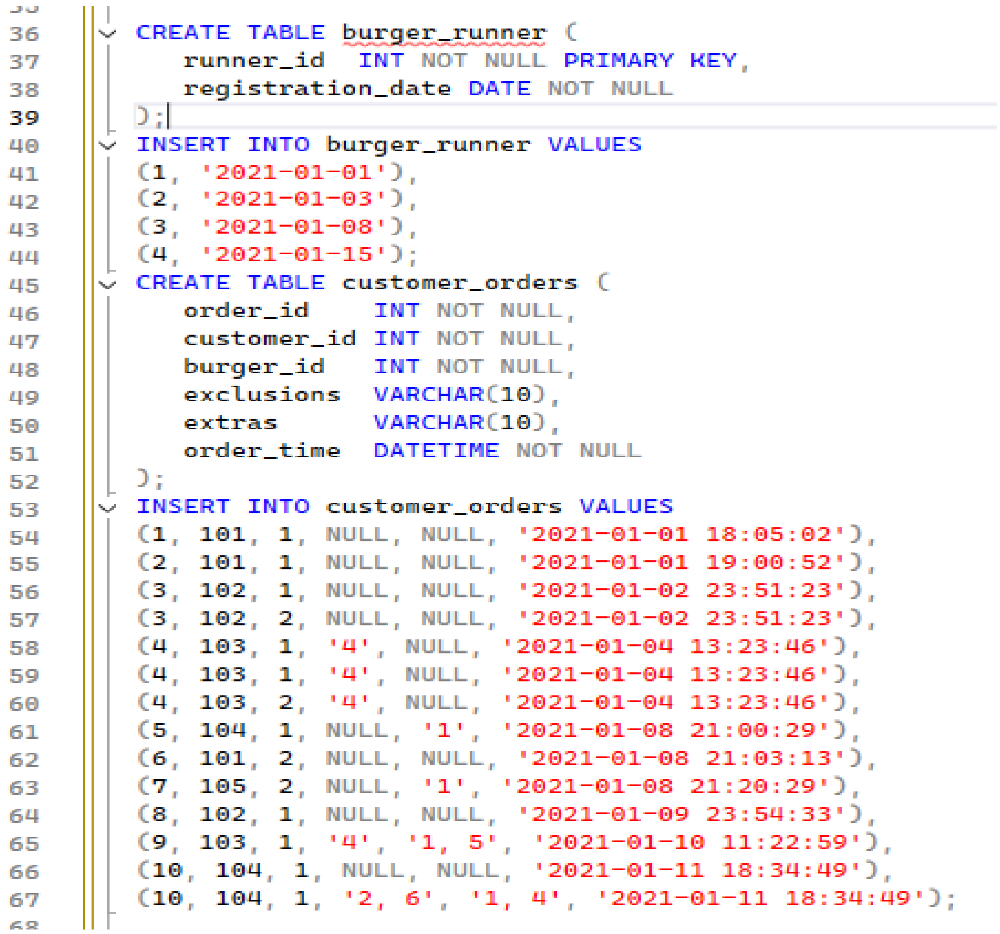
burger_runner	
runner_id	int
registration_date	date

customer_orders	
order_id	int
customer_id	int
burger_id	int
exclusions	varchar
extras	varchar
order_time	timestamp

TABLE CREATION AND INSERTION OF VALUES



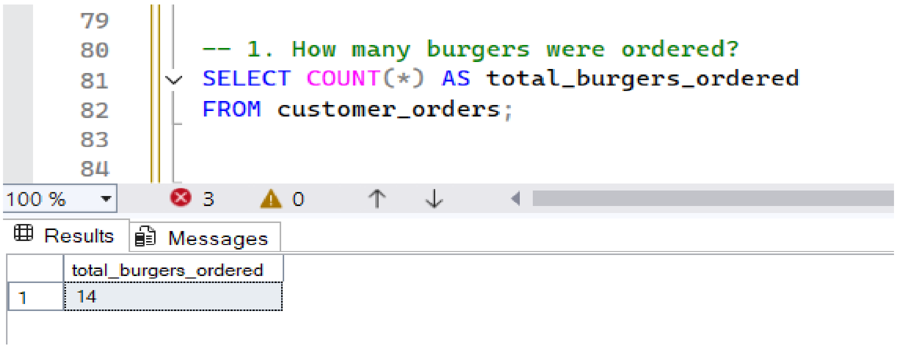
```sql
CREATE DATABASE burger;
-- DROP DATABASE burger;
USE burger;
CREATE TABLE burger_names (
    burger_id   INT NOT NULL PRIMARY KEY,
    burger_name VARCHAR(10) NOT NULL
);

INSERT INTO burger_names (burger_id, burger_name) VALUES
(1, 'Meatlovers'),
(2, 'Vegetarian');

CREATE TABLE runner_orders (
    order_id     INT NOT NULL PRIMARY KEY,-
    runner_id    INT NOT NULL,
    pickup_time  DATETIME,
    distance     VARCHAR(7),
    duration     VARCHAR(10),
    cancellation VARCHAR(23)
);

INSERT INTO runner_orders VALUES
(1, 1, '2021-01-01 18:15:34', '20km', '32 minutes', NULL),
(2, 1, '2021-01-01 19:10:54', '20km', '27 minutes', NULL),
(3, 1, '2021-01-03 00:12:37', '13.4km', '20 mins', NULL),
(4, 2, '2021-01-04 13:53:03', '23.4', '40', NULL),
(5, 3, '2021-01-08 21:10:57', '10', '15', NULL),
(6, 3, NULL, NULL, NULL, 'Restaurant Cancellation'),
(7, 2, '2021-01-08 21:30:45', '25km', '25mins', NULL),
(8, 2, '2021-01-10 00:15:02', '23.4 km', '15 minute', NULL),
(9, 2, NULL, NULL, NULL, 'Customer Cancellation'),
(10, 1, '2021-01-11 18:50:20', '10km', '10minutes', NULL);
```



```sql
CREATE TABLE burger_runner (
    runner_id  INT NOT NULL PRIMARY KEY,
    registration_date DATE NOT NULL
);
INSERT INTO burger_runner VALUES
(1, '2021-01-01'),
(2, '2021-01-03'),
(3, '2021-01-08'),
(4, '2021-01-15');
CREATE TABLE customer_orders (
    order_id    INT NOT NULL,
    customer_id INT NOT NULL,
    burger_id   INT NOT NULL,
    exclusions  VARCHAR(10),
    extras      VARCHAR(10),
    order_time  DATETIME NOT NULL
);
INSERT INTO customer_orders VALUES
(1, 101, 1, NULL, NULL, '2021-01-01 18:05:02'),
(2, 101, 1, NULL, NULL, '2021-01-01 19:00:52'),
(3, 102, 1, NULL, NULL, '2021-01-02 23:51:23'),
(3, 102, 2, NULL, NULL, '2021-01-02 23:51:23'),
(4, 103, 1, '4', NULL, '2021-01-04 13:23:46'),
(4, 103, 1, '4', NULL, '2021-01-04 13:23:46'),
(4, 103, 2, '4', NULL, '2021-01-04 13:23:46'),
(5, 104, 1, NULL, '1', '2021-01-08 21:00:29'),
(6, 101, 2, NULL, NULL, '2021-01-08 21:03:13'),
(7, 105, 2, NULL, '1', '2021-01-08 21:20:29'),
(8, 102, 1, NULL, NULL, '2021-01-09 23:54:33'),
(9, 103, 1, '4', '1, 5', '2021-01-10 11:22:59'),
(10, 104, 1, NULL, NULL, '2021-01-11 18:34:49'),
(10, 104, 1, '2, 6', '1, 4', '2021-01-11 18:34:49');
```

CASE STUDY QUESTIONS

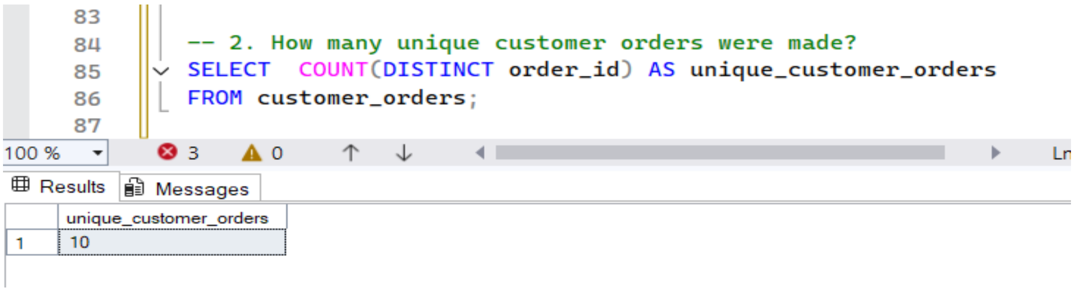
1. How many burgers were ordered?



```sql
-- 1. How many burgers were ordered?
SELECT COUNT(*) AS total_burgers_ordered
FROM customer_orders;
```

	total_burgers_ordered
1	14

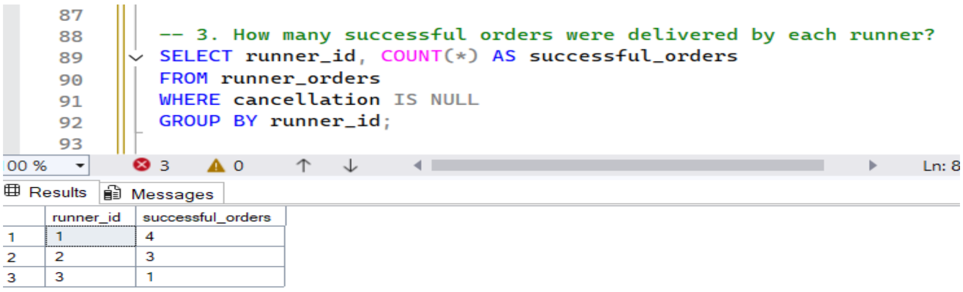
2. How many unique customer orders were made?



```sql
-- 2. How many unique customer orders were made?
SELECT  COUNT(DISTINCT order_id) AS unique_customer_orders
FROM customer_orders;
```

	unique_customer_orders
1	10

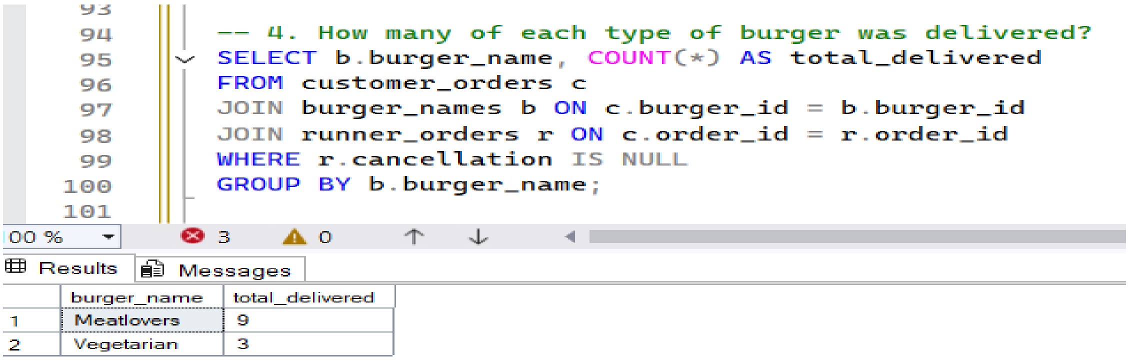
3. How many successful orders were delivered by each runner?



```sql
-- 3. How many successful orders were delivered by each runner?
SELECT runner_id, COUNT(*) AS successful_orders
FROM runner_orders
WHERE cancellation IS NULL
GROUP BY runner_id;
```

	runner_id	successful_orders
1	1	4
2	2	3
3	3	1

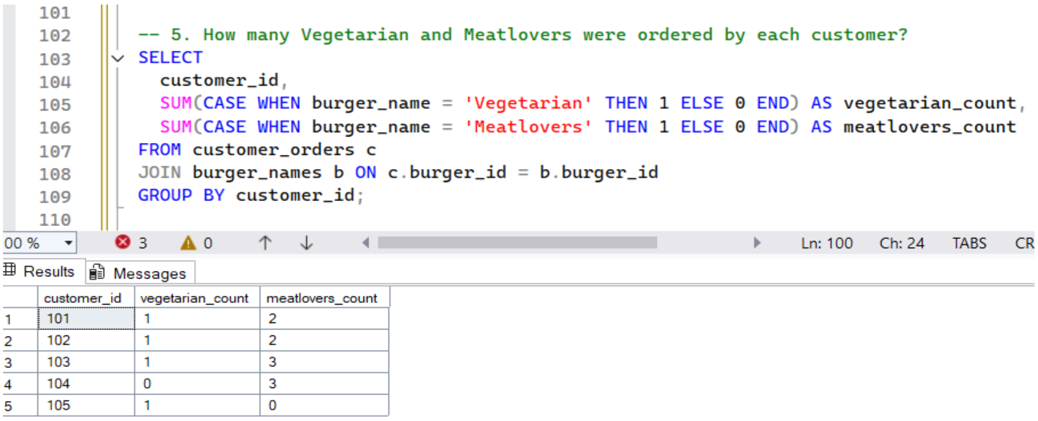
4. How many of each type of burger was delivered?



```sql
-- 4. How many of each type of burger was delivered?
SELECT b.burger_name, COUNT(*) AS total_delivered
FROM customer_orders c
JOIN burger_names b ON c.burger_id = b.burger_id
JOIN runner_orders r ON c.order_id = r.order_id
WHERE r.cancellation IS NULL
GROUP BY b.burger_name;
```

	burger_name	total_delivered
1	Meatlovers	9
2	Vegetarian	3

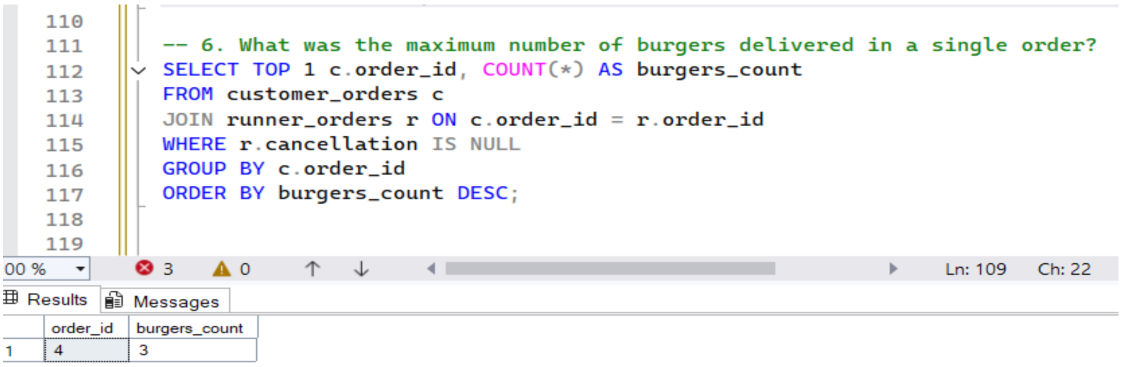
5. How many Vegetarian and Meatlovers were ordered by each customer?



```sql
-- 5. How many Vegetarian and Meatlovers were ordered by each customer?
SELECT
  customer_id,
  SUM(CASE WHEN burger_name = 'Vegetarian' THEN 1 ELSE 0 END) AS vegetarian_count,
  SUM(CASE WHEN burger_name = 'Meatlovers' THEN 1 ELSE 0 END) AS meatlovers_count
FROM customer_orders c
JOIN burger_names b ON c.burger_id = b.burger_id
GROUP BY customer_id;
```

	customer_id	vegetarian_count	meatlovers_count
1	101	1	2
2	102	1	2
3	103	1	3
4	104	0	3
5	105	1	0

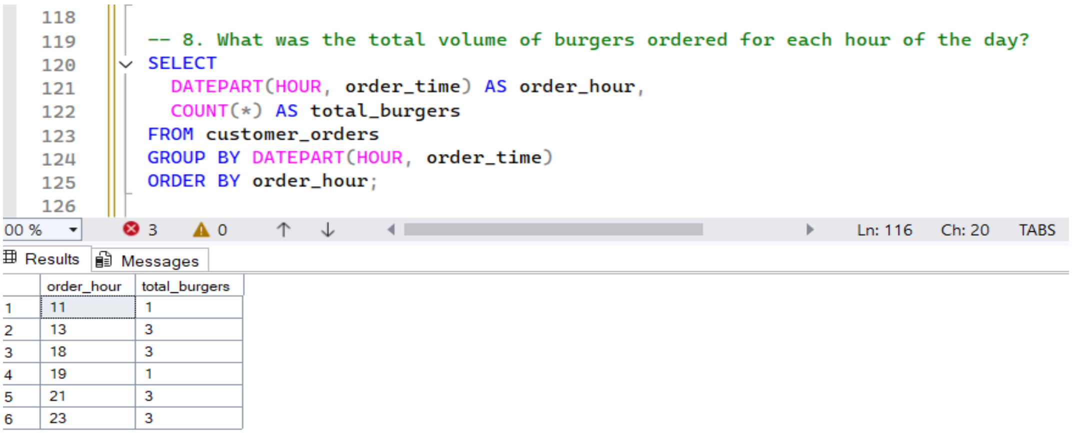
6. What was the maximum number of burgers delivered in a single order?



```sql
-- 6. What was the maximum number of burgers delivered in a single order?
SELECT TOP 1 c.order_id, COUNT(*) AS burgers_count
FROM customer_orders c
JOIN runner_orders r ON c.order_id = r.order_id
WHERE r.cancellation IS NULL
GROUP BY c.order_id
ORDER BY burgers_count DESC;
```

	order_id	burgers_count
1	4	3

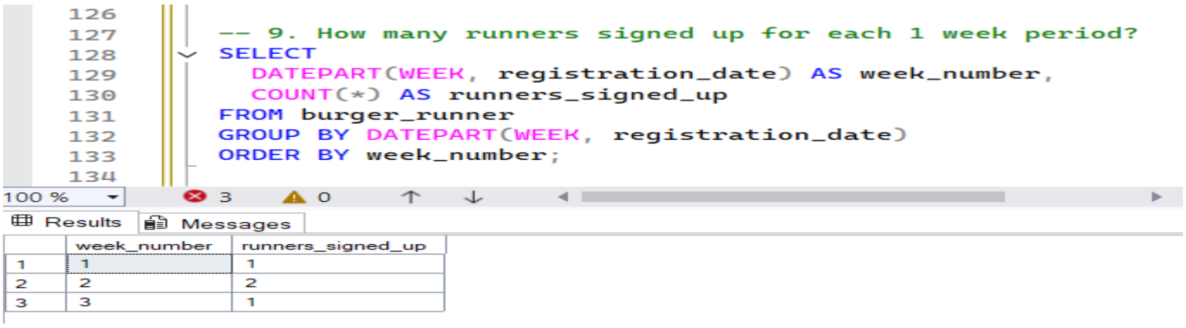
8. What was the total volume of burgers ordered for each hour of the day?



```sql

-- 8. What was the total volume of burgers ordered for each hour of the day?
SELECT
  DATEPART(HOUR, order_time) AS order_hour,
  COUNT(*) AS total_burgers
FROM customer_orders
GROUP BY DATEPART(HOUR, order_time)
ORDER BY order_hour;

```

	order_hour	total_burgers
1	11	1
2	13	3
3	18	3
4	19	1
5	21	3
6	23	3

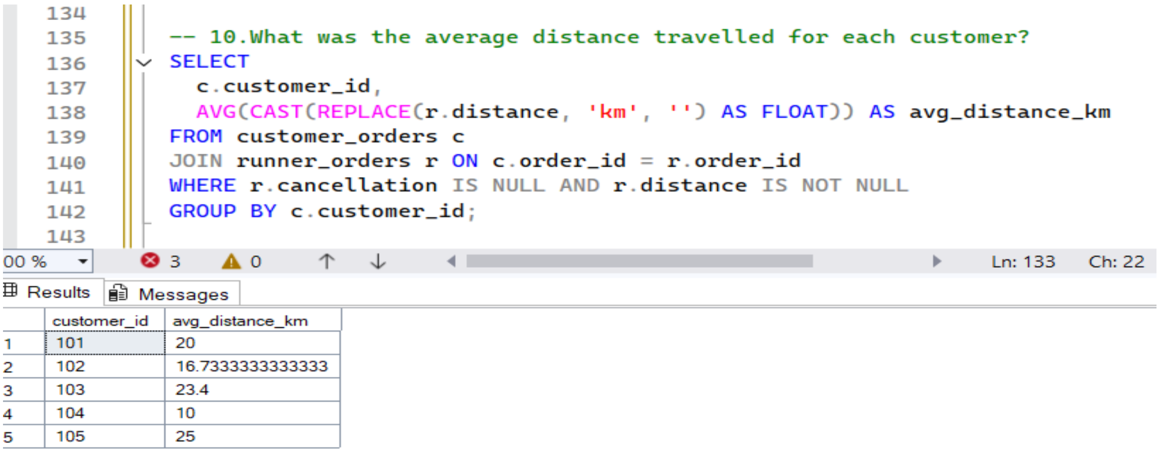
9. How many runners signed up for each 1 week period?



```sql

-- 9. How many runners signed up for each 1 week period?
SELECT
  DATEPART(WEEK, registration_date) AS week_number,
  COUNT(*) AS runners_signed_up
FROM burger_runner
GROUP BY DATEPART(WEEK, registration_date)
ORDER BY week_number;

```

	week_number	runners_signed_up
1	1	1
2	2	2
3	3	1

10.What was the average distance travelled for each customer?



```sql

-- 10.What was the average distance travelled for each customer?
SELECT
  c.customer_id,
  AVG(CAST(REPLACE(r.distance, 'km', '') AS FLOAT)) AS avg_distance_km
FROM customer_orders c
JOIN runner_orders r ON c.order_id = r.order_id
WHERE r.cancellation IS NULL AND r.distance IS NOT NULL
GROUP BY c.customer_id;

```

	customer_id	avg_distance_km
1	101	20
2	102	16.7333333333333
3	103	23.4
4	104	10
5	105	25