FLOW Is like this.
HDD<—OS<—DBS<—DBMS(phpMyAdmin) MYSQL/Oracle<—DB<—USER Here
DBS means (Data Base Server) DB means (Data Bases)

DATATYPES IN SQL

NUMERICAL

TYPES OF INTEGER DATATYPE

TINYINT It is used for age

SMALLINT It issued for Weight

MEDIUMINT It is used for pin code

MEDIUMINT It also has 2 types SignedINT & UnsignedINT
SignedINT It store both +ve & -ve No. -128 to 127 (-128,-127,-126…0,1,2…127)
UnsignedINT It store only +ve No. 0 to 256 (0,1,2…256)

INT It used for salary

BIGINT It is used for phone No, salary.

TEXT

TYPES OF STRING DATATYPES

CHAR:- It is used to store fixed length string. Length is predefined while creating table. Now while creating if give CHAR(10) then it will be available 10 later space to store string we can store only 10 laters in it not more than that. And if we store 6 later in it then it means we are wasting 4 later space.

VARCHAR:- It is used to store variable length string. Mean if store ‘hi’ then it will take only 2 later space but if we write ‘Aniket’ then it will take 6 later. Means it gets adjusted by itself depending upon the length of string we are inserting.

TEXT:- It is use to store large amount of textual data like blog post, comment.

MEDIUMTEXT:- It is use to store much large amount of textual data like article.

LONGTEXT:- It is use to store largest amount of textual data like books.

ENUM :- It is data Type used to store predefined values.. we have to provide list of values. So this data type will ensure that only predefined values provided will get store in columns.But at a time it will set only one value in columns. Ex.Gender (Male, Female, Other) so in gender column either Male or Female or Other in respective rows.

SET :- It is data Type used to store predefined values.. we have to provide list of values. So this data type will ensure that only predefined values provided will get store in columns. But at a time it will set more than one values in respective rows within columns. Ex. Hobbies (Singing,Playing,Dancing) so in column (Singing,Playing) or (Playing,Dancing) etc.

Blob:- (Binary Large Object) It store pdf, images, audio, video etc.
It also has 4 types
TIYNYBLOB It store very small data like icon etc.
BLOB It store small data like small images, small audio files. etc.
MEDIUMBLOB It store data like HD images, medium audio files. etc.
LONGBLOB It stores data like HD videos and 4K images.

Note:- while adding this unstructured files in it we have to call one function I.e. Load Files()

Spatial Datatypes

Geometry:- It is used to store latitude and longitude like info.

Other Datatypes

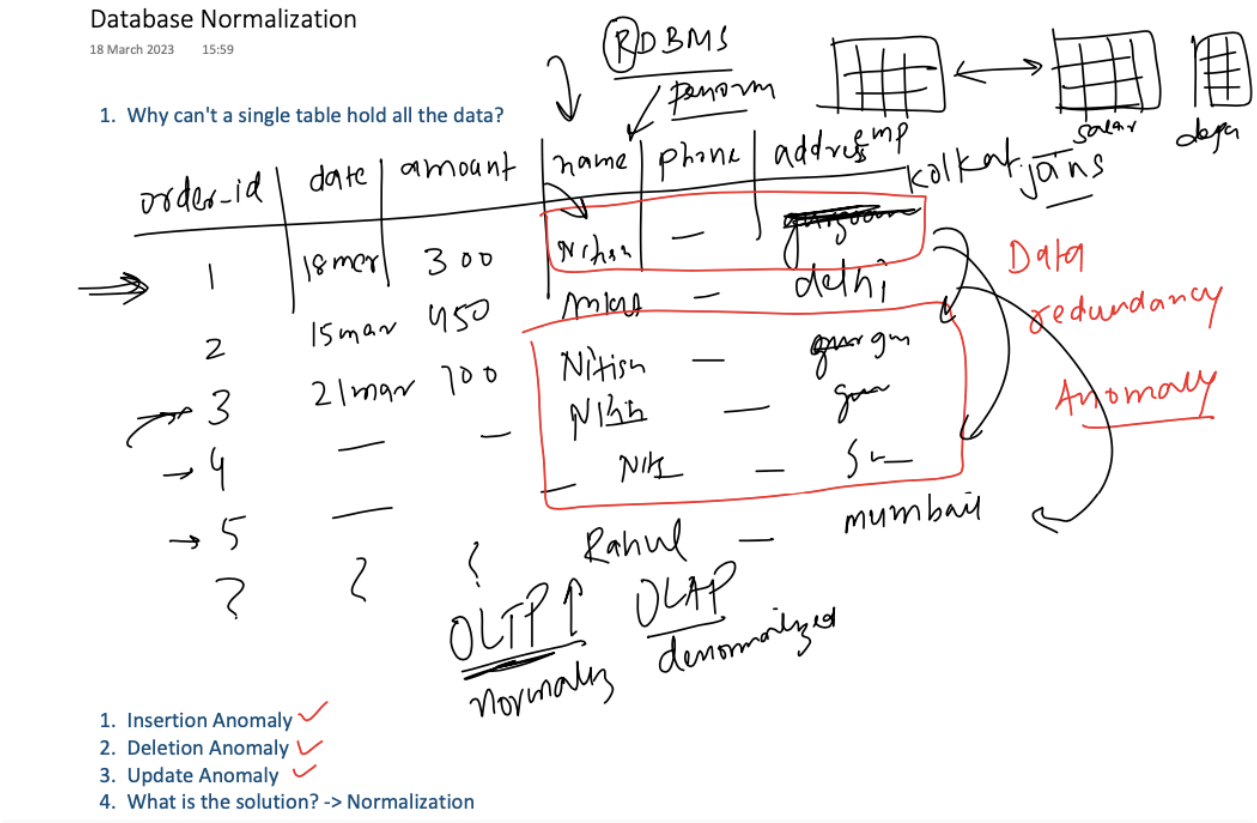
JSON:- It is used to store data like key-value.

TIME

TYPES TEMPORAL DATATYPE

DATE:- It is used to store DOB or DOD like things. (YYYY-MM-DD)
TIME:- It is Used to store time in this format(HH-MM-SS) used to set alarm.
DATETIME:- It will give both date and time (YYYY-MM-DD HH-MM-SS). ORDER DETALS
TIMESTAMP:-It is same as DATETIME with very less difference between both.
YEAR:- It is used to store pass out year like things.(YYYY) OR (YY)

DATABASE NORMALIZATION



Database normalization is a process used to organize data in a database to reduce data redundancy and dependency. The goal of normalization is to ensure that each piece of data is stored in one place, in a structured way, to minimize the risk of inconsistencies and improve the overall efficiency and usability of the database.

There are several levels of database normalization, each with its own set of rules and guidelines. The most commonly used levels of normalization are:

First Normal Form (1NF): This level requires that all data in a table is stored in a way that each column contains only atomic (indivisible) values, and there are no repeating groups or arrays.
Rules:- In particular column and every row must contain only one value.

Once we set data type of particular column(attribute) it should not be change in Between.

There should be unique name of every column(attribute).

Second Normal Form (2NF): This level requires that each non-key attribute in a table is dependent on the entire primary key, not just a part of it.
Rules:-

It should be in 1st Normal form.

It should not contain any partial dependancies.

Partial dependency occurs when a non-key attribute is dependent on only a part of the primary key instead of the entire key.

Third Normal Form (3NF): This level requires that each non-key attribute in a table is dependent only on the primary key and not on any other non-key attributes.

There are higher levels of normalization, such as Fourth Normal Form (4NF) and Fifth Normal Form (5NF), but they are less commonly used in practice.

Rules:-

It should be in 2nd normal form.

There should not be transitive dependancies.

A transitive dependency exists when a non-key attribute depends on another non-key attribute, which is not a part of the primary key.

Student Table

Roll No	Name	Branch	Email
1	Nitish Sing	CSE	Nitish@gmail.com
2	Ankit Sharma	EEE	Ankit@gmail.com
3	Neha Verma	ME	Neha@gmail.com

All Possible Keys

Single

Roll No (Yes)
Email (Yes)

Double
Roll No + Name (Yes)
Roll No + Branch (Yes)
Roll No + Email (Yes)

Email + Name (Yes)
Email + Branch (Yes)

Name + Branch (No It would not be a key)

Triple
Roll No + Name + Branch (Yes)
Roll No + Name + Email (Yes)

Email + Branch + Name (Yes)

Combination of all 4
Roll No + Name + Branch + Email (Yes)

Total 11 keys are possible from above table.

KEYS (Total 9 keys)

Super Key :- (All possible Combinations) [JANTA]

Single
Roll No (Yes)
Email (Yes)
Double
Roll No + Name (Yes)
Roll No + Branch (Yes)
Roll No + Email (Yes)
Email + Name (Yes)
Email + Branch (Yes)
Triple
Roll No + Name + Branch (Yes)
Roll No + Name + Email (Yes)
Email + Branch + Name (Yes)
Combination of all 4
Roll No + Name + Branch + Email (Yes)

Candidate key [Umedwar]

It is minimal super Key i.e. it has redundant attribute. Smallest set of tuple which uniquely identifies row.

Singles
Roll No (Yes)
Email (Yes)

Primary Key [Winning Person]

{ Compulsory Criteria
Not Null
Unique (It should not be duplicated)
Good to have Criteria
Numerical
Small
It should be Constant **}**

Single
Roll No (Yes)

Email (No)
Both fulfilled Compulsory Criteria but Email failed in 2nd & 3rd of Good to have Criteria.

Alternate Key (AK=CK-PK)

Single
Email (Yes)

Composite Key

It is made from 2 or more than 2 attribute. It is used when single attribute is not sufficient to uniquely identify each row.

Student
Sid Name Email Phone

Course
Cid Name Price Instructor

Enrolment
Sid Cid Date Payment

Sid + Cid (Yes)

Surrogate Key

We add one column from our side which act as key that key is called as Surrogate Key

Student

Name Branch CGPA
Sid (Yes)

– – – – – – – ->

Student

Sid Name Branch CGPA

Foreign Key

It is PK from one table is used to establish a relationship with another table.

Student
Sid Name Email Phone
PK

Course
Cid Name Price Instructor
PK

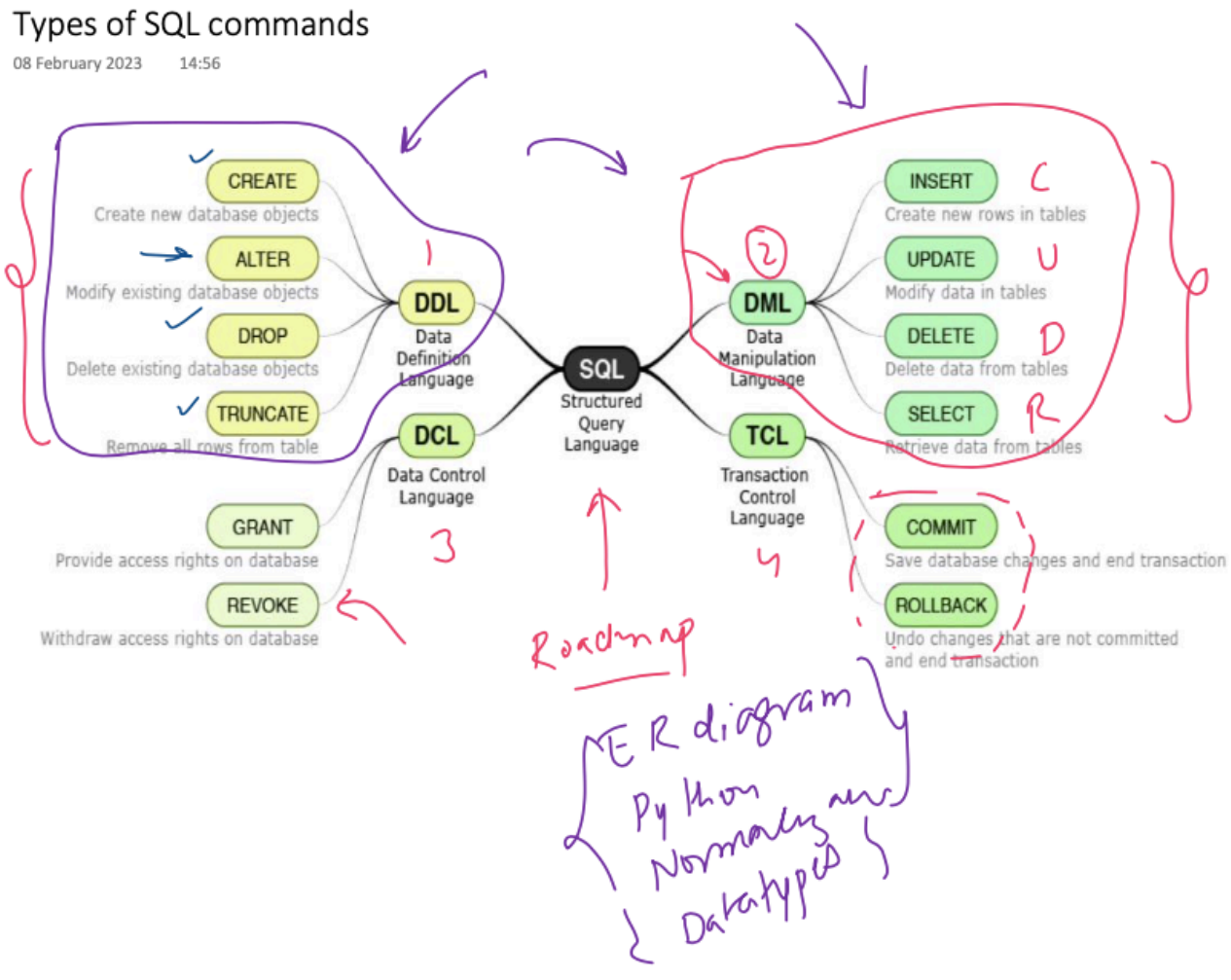
Enrolment
Sid Cid Date Payment
FK **FK**

Compound Key
Unique key

SQL

What is SQL?

SQL (Structured Query Language) is a programming language used for managing and manipulating data in relational databases. It allows you to insert, update, retrieve, and delete data in a database. It is widely used for data management in many applications, websites, and businesses. In simple terms, SQL is used to communicate with and control databases.



DDL Commands

1. CREATE 2. DROP 3. TRUNCATE 4. ALTER

DDL Commands for Databases

1. CREATE 2. DROP

DDL Commands for Tables

1. CREATE 2. DROP 3. TRUNCATE 4. ALTER (To Rename the Table Name)

DDL Commands for Columns in the table then..

1. ALTER (To 'Add' Column (**ADD COLUMN**) or To Delete Column(**DROP COLUMN**) or To Rename (**RENAME COLUMN**) the Column) from the table, use 'ALTER'.

ALTER TABLE Command

The "**ALTER TABLE**" statement in SQL is used to modify the structure of an existing table. Some of the things that can be done using the ALTER TABLE statement include

Data Integrity

Data integrity in databases refers to the accuracy, completeness, and consistency of the data stored in a database. It is a measure of the reliability and trustworthiness of the data and ensures that the data in a database is protected from errors, corruption, or unauthorised changes.

There are various methods used to ensure data integrity, including:

Constraints:
Constraints in databases are rules or conditions that must be met for data to be inserted, updated, or deleted in a database table. They are used to enforce the integrity of the data stored in a database and to prevent data from becoming inconsistent or corrupted.

Transactions: a sequence of database operations that are treated as a single unit of work.

Normalisation: a design technique that minimises data redundancy and ensures data consistency by organising data into separate tables.

Constraints in MySQL

Constraints in databases are rules or conditions that must be met for data to be inserted, updated, or deleted in a database table. They are used to enforce the integrity of the data stored in a database and to prevent data from becoming inconsistent or corrupted.

NOT NULL

UNIQUE(combo)

-> Another way of creating constraint

PRIMARY KEY

AUTO INCREMENT

CHECK

DEFAULT

FOREIGN KEY

Referential Actions

RESTRICT

CASCADE

SET NULL

SET DEFAULT

```sql
CREATE TABLE ticket(
    ticket_id INTEGER PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    travel_date DATETIME DEFAULT CURRENT_TIMESTAMP
)
```

Run SQL query/queries on database campusx:

```sql
CREATE TABLE users(
    user_id INTEGER NOT NULL,
    name VARCHAR(255) NOT NULL,
    email VARCHAR(255) NOT NULL,
    password VARCHAR(255) NOT NULL,

    CONSTRAINT users_email_unique UNIQUE (name,email),
    CONSTRAINT users_pk PRIMARY KEY (user_id,name)
)
```

Editing and Deleting Constraints For this also use Alter

ADD

ALTER TABLE customers ADD CONSTRAINT customer_age_check CHECK (age > 13)

DROP

(ALTER TABLE customers DROP CONSTRAINT customer age check)

~~**EDIT**~~ (It is Not possible)

DML Commands

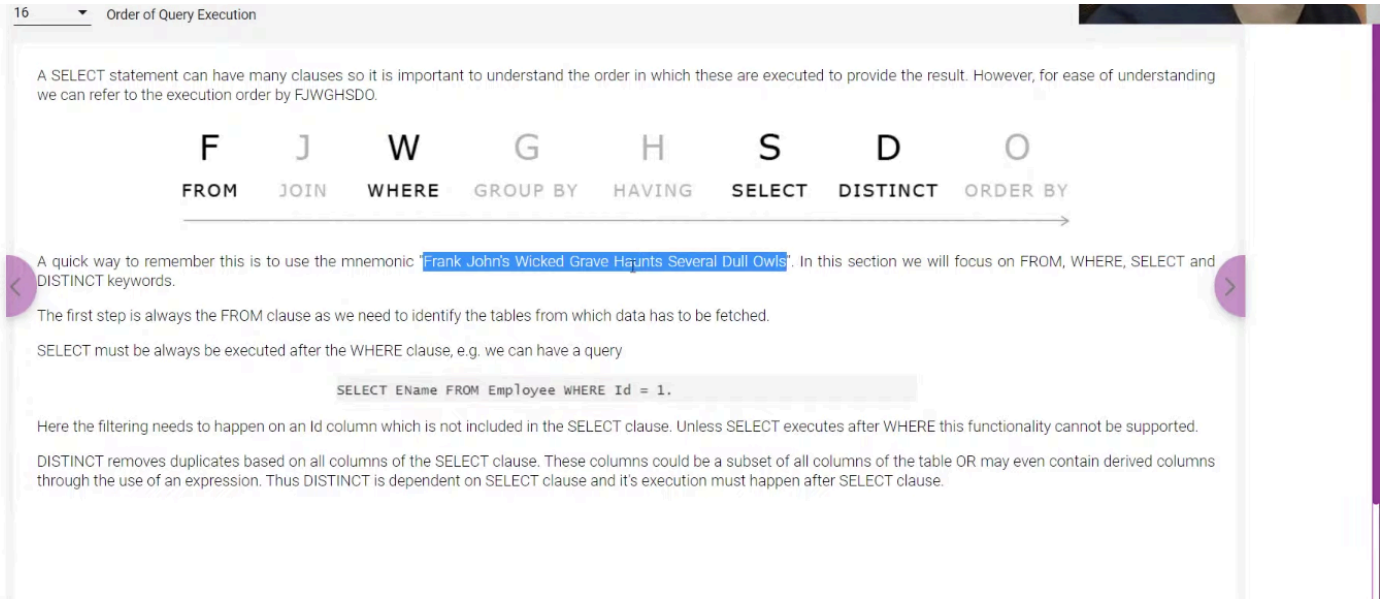
INSERT(VALUES)	SELECT	UPDATE(SET)	DELETE(FROM...WHERE...)
C	R	U	D

Note:-
UPDATE AND DELETE Command Without where condition are very risky.

Note:-
DDL here we change the structure
DDL statement can not be rollback.

DML statement can be rollback.
DML here we change the data.

ORDER OF EXECUTION



SQL Functions

BUILTIN FUNCTION. **USER DEFINED FUNCTION.**

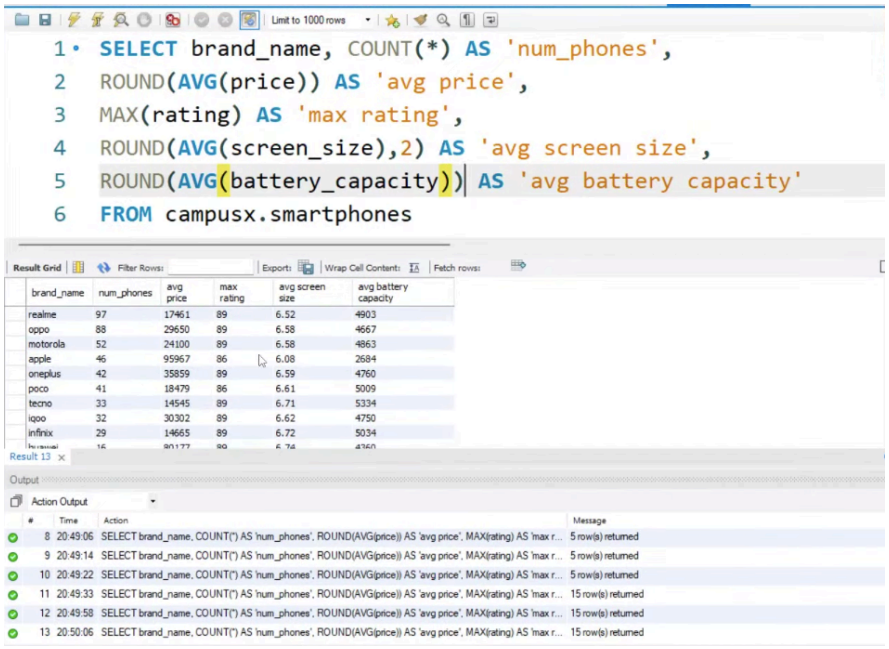
SCALAR ***AGGREGATE***

EX. ROUND
ABS
CEIL
FLOOR

EX. SUM
MIN/MAX
AVG
COUNT
STD/ VARIENCE

TO DO FILTRATION ON "SELECT" WE HAVE "WHERE"
AND
TO DO FILTRATION ON "GROUP BY" WE HAVE "HAVING"

GROUPING + SORTING



GROUP BY ON MORE THAN 2 COLUMNS

```
1  SELECT brand_name,
2  processor_brand,
3  COUNT(*) AS 'num phones',
4  ROUND(AVG(primary_camera_rear)) AS 'avg camera resolution'
5  FROM campusx.smartphones
6  GROUP BY brand_name,processor_brand
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

brand_name	processor_brand	num phones	avg camera resolution
samsung	snapdragon	36	71
samsung	unisoc	3	35
sharp	snapdragon	1	48
sony	helio	1	13
sony	snapdragon	8	17
tcl	helio	1	13
tecno		1	13
tecno	dimensity	6	55
tecno	helio	23	40

Result 3

Output

Action Output

#	Time	Action	Message
16	20:57:29	SELECT has_5g, AVG(price) AS 'avg price', AVG(rating) AS 'rating' FROM campusx.smartphones GROUP BY...	2 row(s) returned
17	20:58:04	SELECT * FROM campusx.smartphones LIMIT 0, 1000	980 row(s) returned
18	20:58:25	SELECT fast_charging_available, AVG(price) AS 'avg price', AVG(rating) AS 'rating' FROM campusx.smartpho...	2 row(s) returned
19	20:59:21	SELECT extended_memory_available, AVG(price) AS 'avg price', AVG(rating) AS 'rating' FROM campusx.sma...	2 row(s) returned
20	21:07:14	SELECT brand_name, processor_brand, COUNT(*) AS 'num phones', AVG(primary_camera_rear) AS 'avg cam...	94 row(s) returned
21	21:07:29	SELECT brand_name, processor_brand, COUNT(*) AS 'num phones', ROUND(AVG(primary_camera_rear)) AS ...	94 row(s) returned

QUERY

Find the top 3 brands with the highest avg ram that have a refresh rate of at least 90 Hz and fast charging available and dont consider brands which have less than 10 phones

```
SELECT  brand_name,
AVG(ram capacity) AS 'avg ram
FROM campusx.smartphones
WHERE refresh_rate > 90 AND fast_charging available =1
GROUP BY brand name
HAVING COUNT(*) > 10
ORDER BY
'avggram'
DESC LIMIT 3
```

QUERY

Find the avg price of all the phone brands with avg rating > 70 and num phones more than 10 among all 5g enabled phones

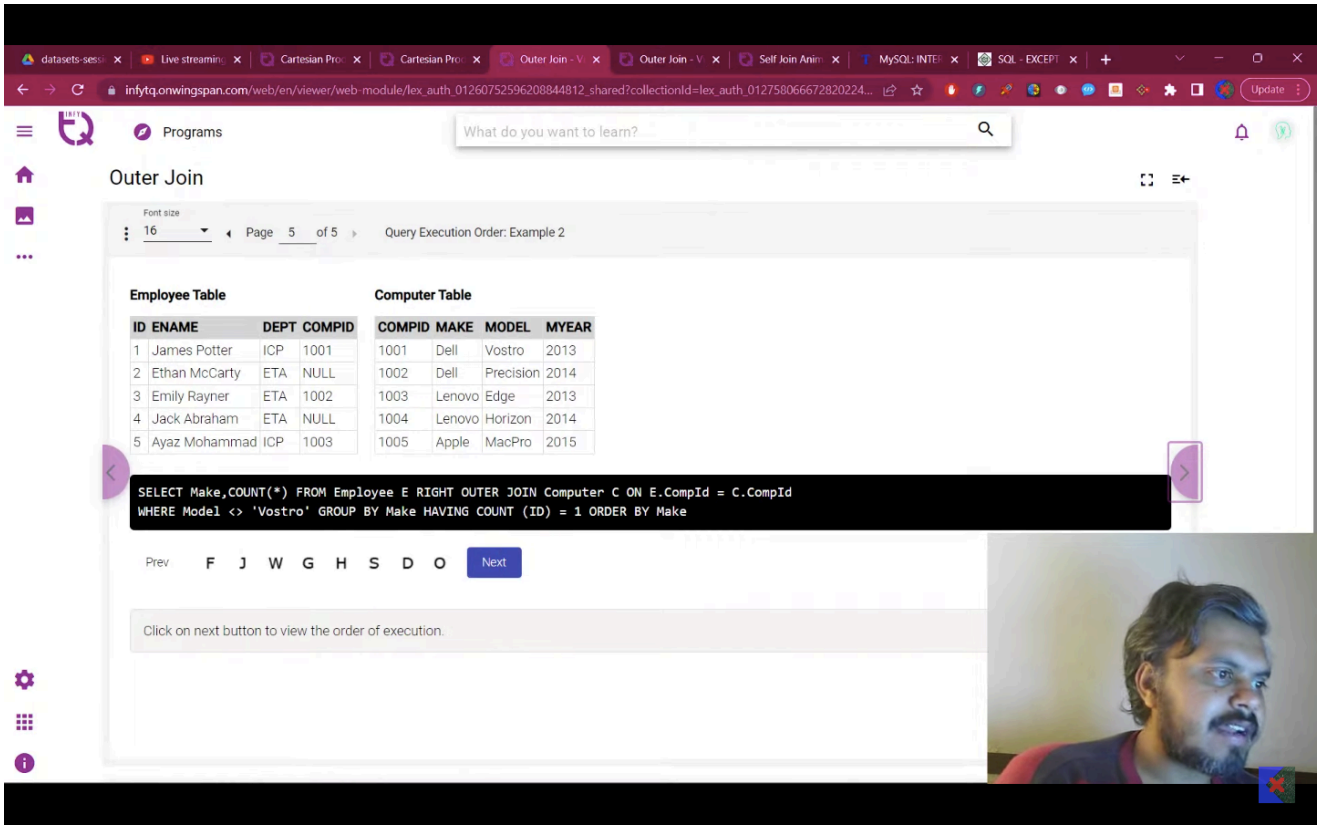
```
SELECT brand_name, AVG (price) AS 'avg_price'
FROM campusx.smartphones
WHERE has 5g =
'True'
GROUP BY brand name
HAVING AVG (rating) › 70 AND COUNT(*) > 10
```

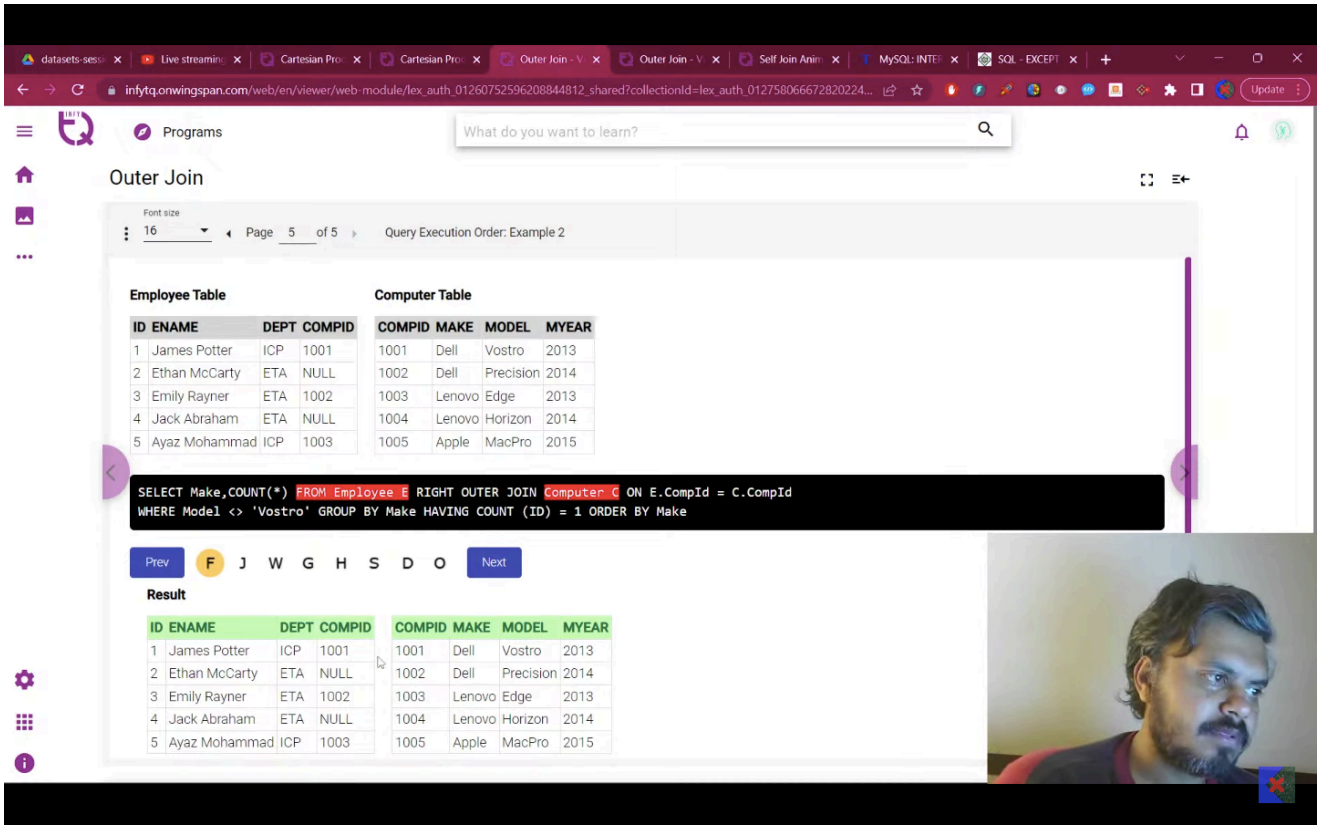
TOP 5 BATSMAN WHOES STRIKE RATE IS HIGHEST WITH CONDITION THAT BATSMAN PLAYED MORE THAN 1000 BALLS.

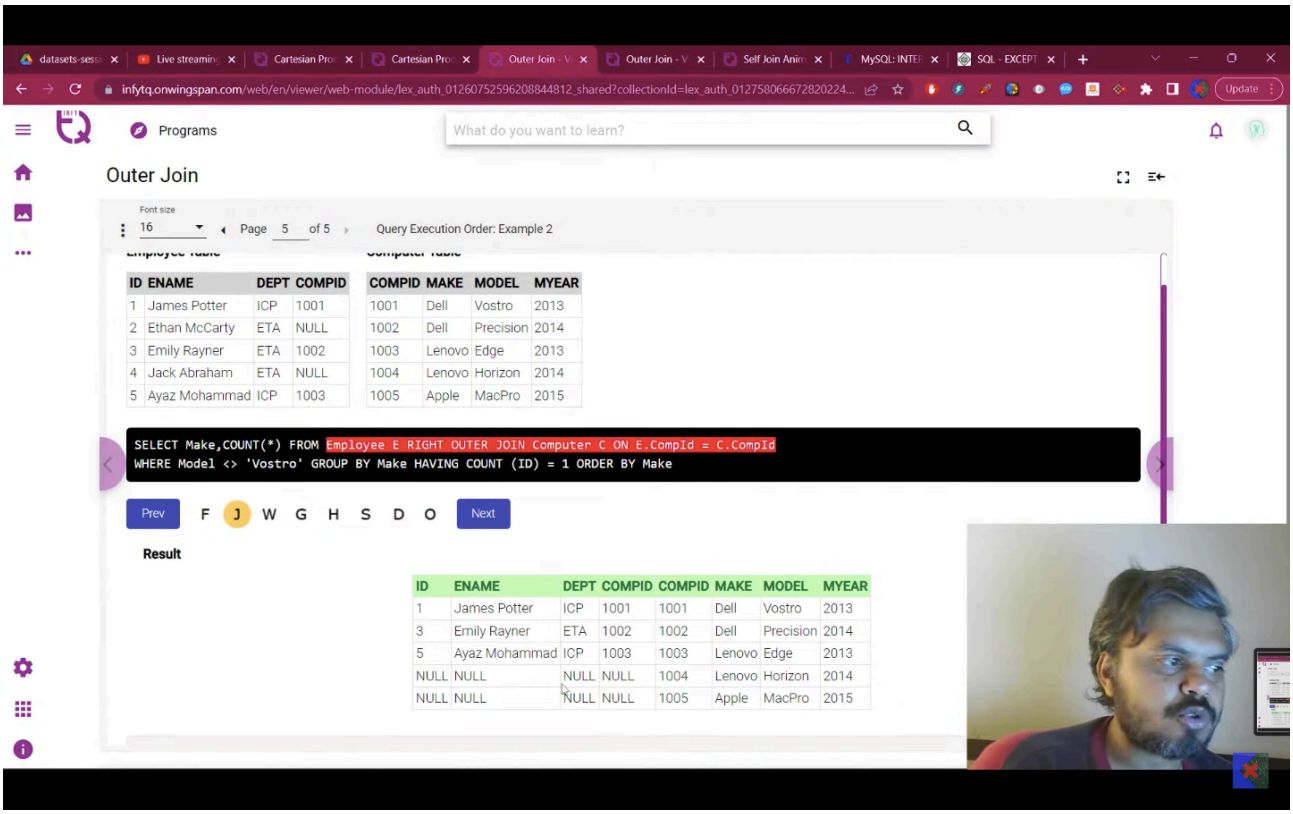
```
SELECT batter, SUM (batsman_run), COUNT (batsman_run),
ROUND((SUM(batsman_run)/COUNT (batsman_run)) * 100,2) AS 'sr'
FROM campusx.ipl
GROUP BY batter
HAVING COUNT (batsman_run) >
```

JOINS

FLOW OF QUERY



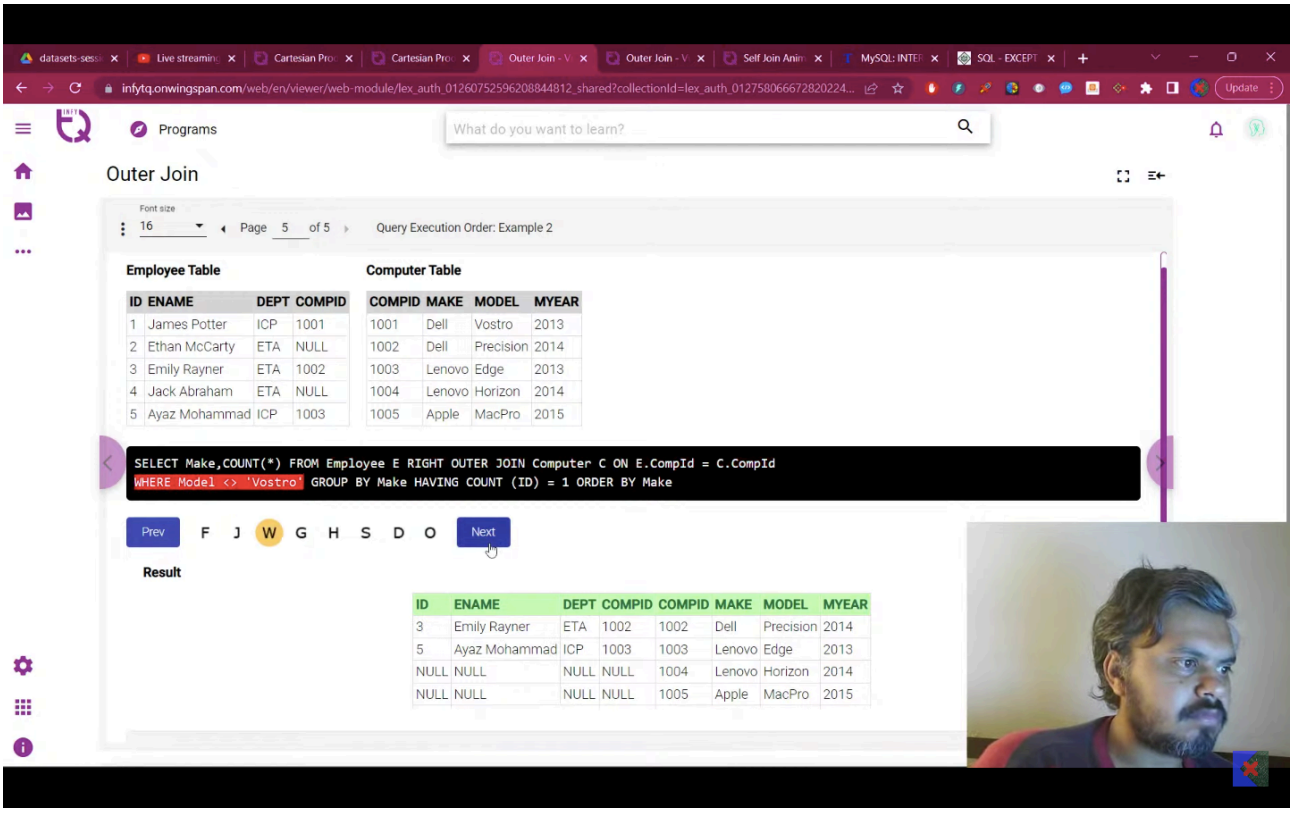




ID	ENAME	DEPT	COMPID
1	James Potter	ICP	1001
2	Ethan McCarty	ETA	NULL
3	Emily Rayner	ETA	1002
4	Jack Abraham	ETA	NULL
5	Ayaz Mohammad	ICP	1003

COMPID	MAKE	MODEL	MYEAR
1001	Dell	Vostro	2013
1002	Dell	Precision	2014
1003	Lenovo	Edge	2013
1004	Lenovo	Horizon	2014
1005	Apple	MacPro	2015

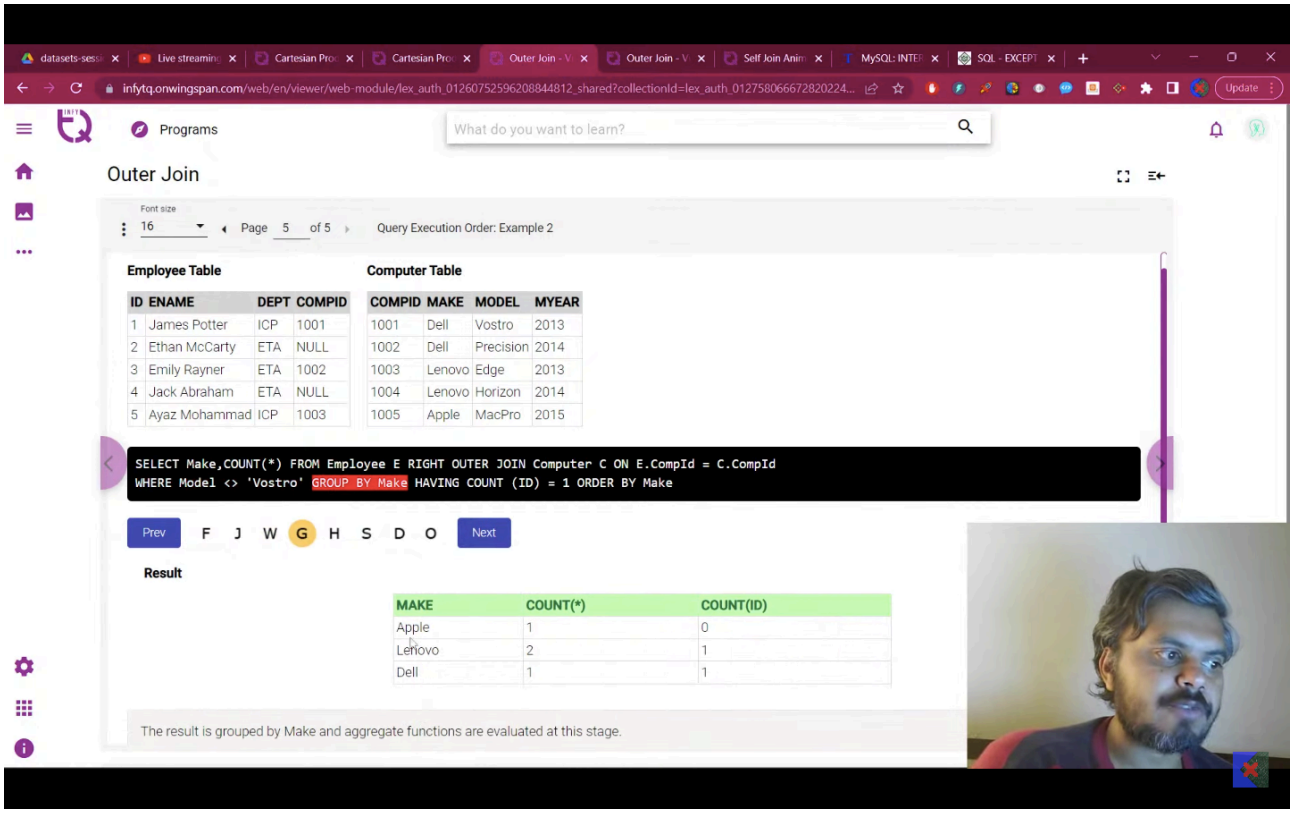
ID	ENAME	DEPT	COMPID	COMPID	MAKE	MODEL	MYEAR
1	James Potter	ICP	1001	1001	Dell	Vostro	2013
3	Emily Rayner	ETA	1002	1002	Dell	Precision	2014
5	Ayaz Mohammad	ICP	1003	1003	Lenovo	Edge	2013
NULL	NULL	NULL	NULL	1004	Lenovo	Horizon	2014
NULL	NULL	NULL	NULL	1005	Apple	MacPro	2015

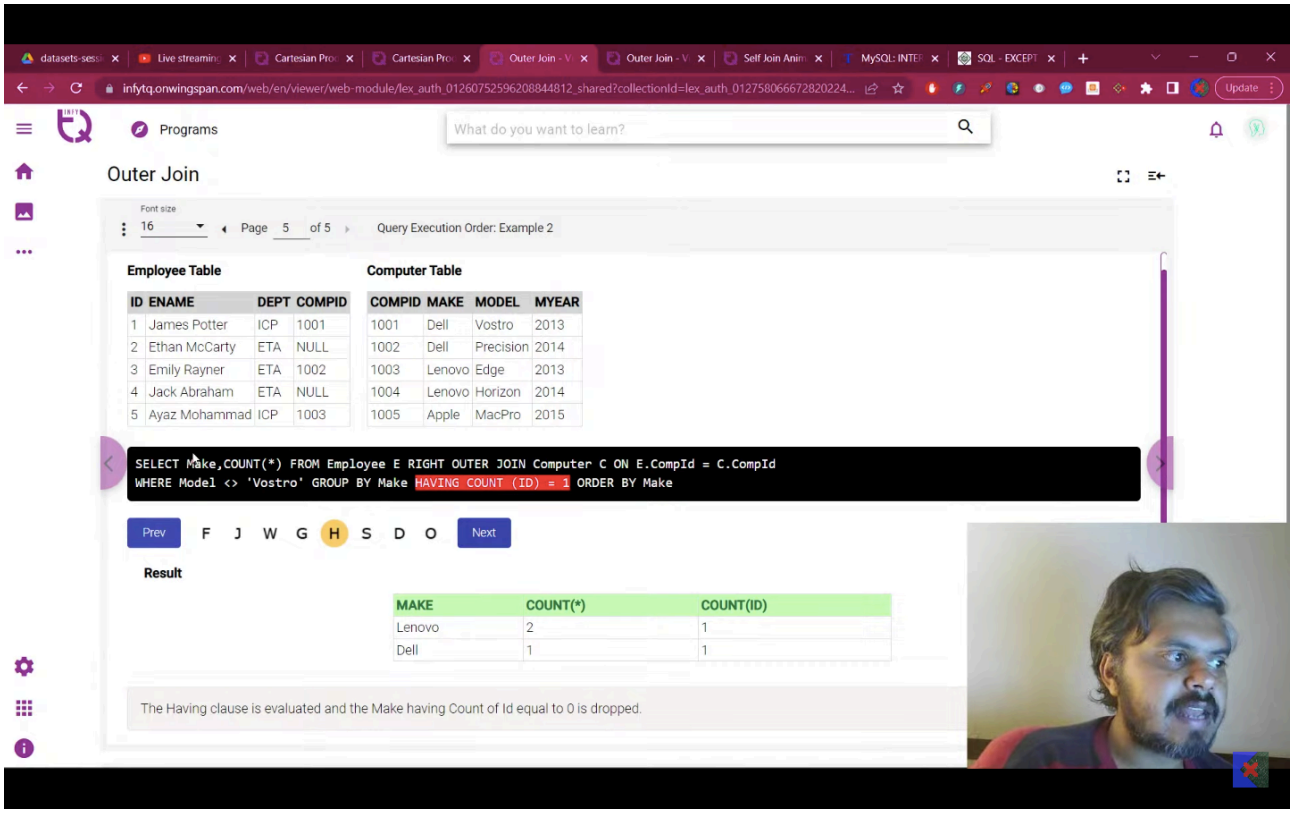


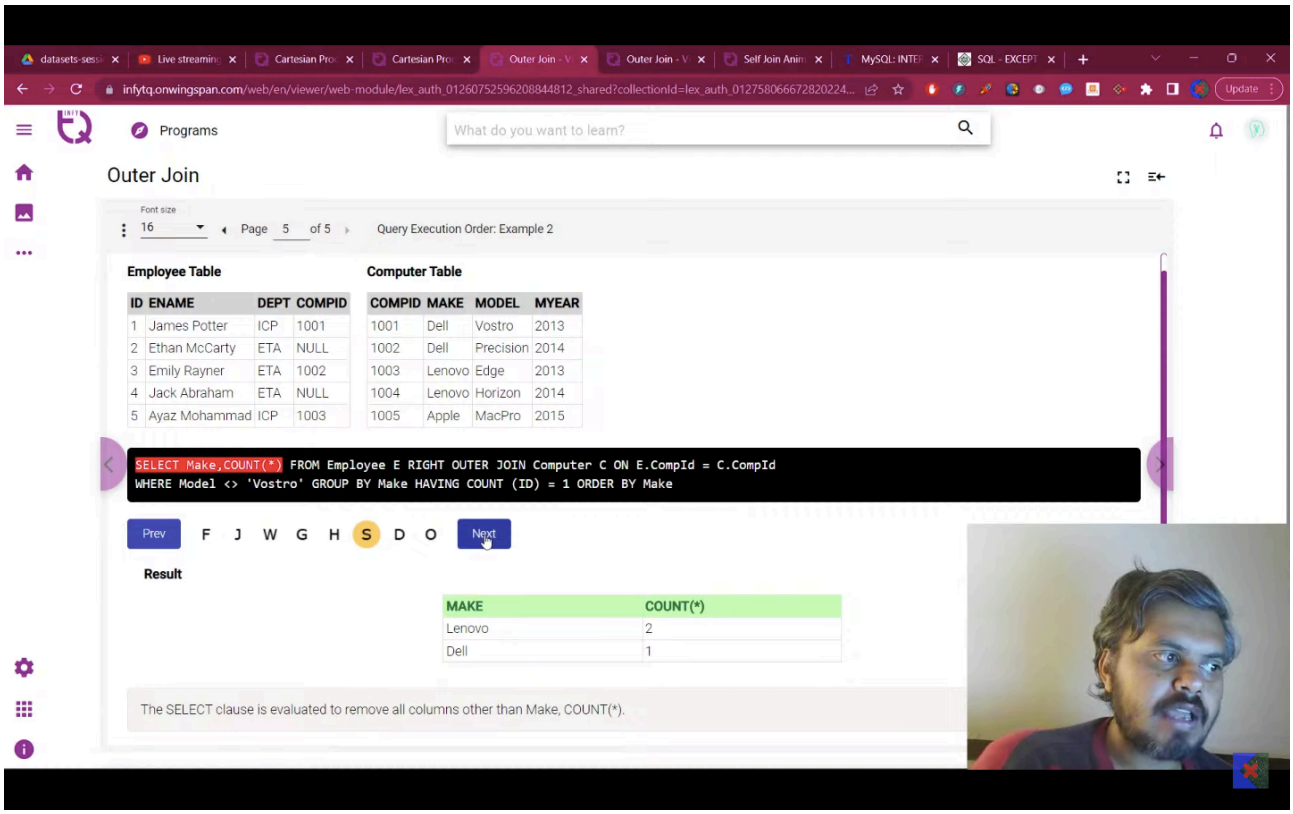
ID	ENAME	DEPT	COMPID
1	James Potter	ICP	1001
2	Ethan McCarty	ETA	NULL
3	Emily Rayner	ETA	1002
4	Jack Abraham	ETA	NULL
5	Ayaz Mohammad	ICP	1003

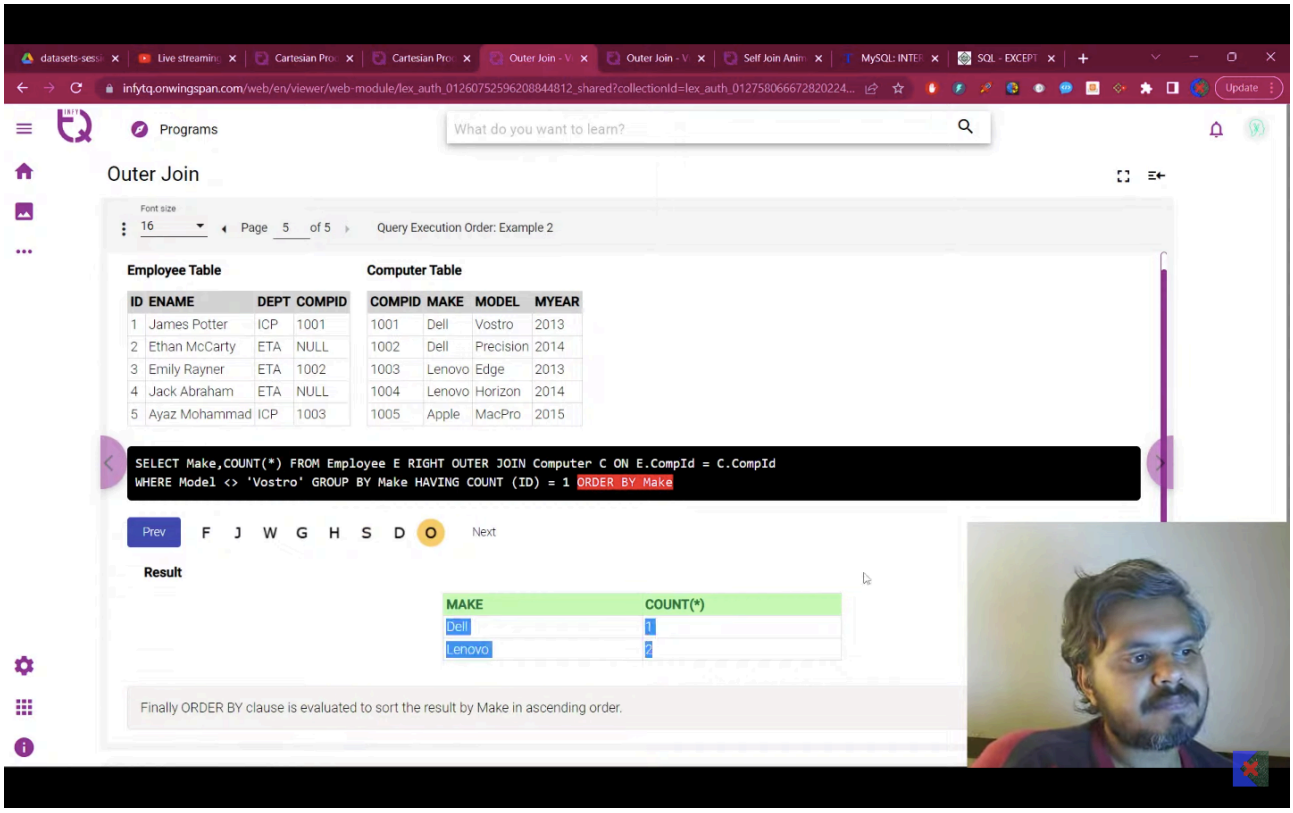
COMPID	MAKE	MODEL	MYEAR
1001	Dell	Vostro	2013
1002	Dell	Precision	2014
1003	Lenovo	Edge	2013
1004	Lenovo	Horizon	2014
1005	Apple	MacPro	2015

ID	ENAME	DEPT	COMPID	COMPID	MAKE	MODEL	MYEAR
3	Emily Rayner	ETA	1002	1002	Dell	Precision	2014
5	Ayaz Mohammad	ICP	1003	1003	Lenovo	Edge	2013
NULL	NULL	NULL	NULL	1004	Lenovo	Horizon	2014
NULL	NULL	NULL	NULL	1005	Apple	MacPro	2015


Programs
What do you want to learn?
Outer Join
Font size 16
Page 5 of 5
Query Execution Order: Example 2
Employee Table
ID ENAME DEPT COMPID
1 James Potter ICP 1001
2 Ethan McCarty ETA NULL
3 Emily Rayner ETA 1002
4 Jack Abraham ETA NULL
5 Ayaz Mohammad ICP 1003
Computer Table
COMPID MAKE MODEL MYEAR
1001 Dell Vostro 2013
1002 Dell Precision 2014
1003 Lenovo Edge 2013
1004 Lenovo Horizon 2014
1005 Apple MacPro 2015
SELECT Make,COUNT(*) FROM Employee E RIGHT OUTER JOIN Computer C ON E.CompId = C.CompId
WHERE Model <> 'Vostro' GROUP BY Make HAVING COUNT (ID) = 1 ORDER BY Make
Prev F J W G H S D O Next
Result
MAKE COUNT(*) COUNT(ID)
Apple 1 0
Lenovo 2 1
Dell 1 1
The result is grouped by Make and aggregate functions are evaluated at this stage.


Programs
What do you want to learn?
Outer Join
Font size 16
Page 5 of 5
Query Execution Order: Example 2
Employee Table
ID ENAME DEPT COMPID
1 James Potter ICP 1001
2 Ethan McCarty ETA NULL
3 Emily Rayner ETA 1002
4 Jack Abraham ETA NULL
5 Ayaz Mohammad ICP 1003
Computer Table
COMPID MAKE MODEL MYEAR
1001 Dell Vostro 2013
1002 Dell Precision 2014
1003 Lenovo Edge 2013
1004 Lenovo Horizon 2014
1005 Apple MacPro 2015
SELECT Make,COUNT(*) FROM Employee E RIGHT OUTER JOIN Computer C ON E.CompId = C.CompId
WHERE Model <> 'Vostro' GROUP BY Make HAVING COUNT (ID) = 1 ORDER BY Make
Prev F J W G H S D O Next
Result
MAKE COUNT(*) COUNT(ID)
Lenovo 2 1
Dell 1 1
The Having clause is evaluated and the Make having Count of Id equal to 0 is dropped.


Programs
What do you want to learn?
Outer Join
Font size
16
Page 5 of 5
Query Execution Order: Example 2
Employee Table
ID ENAME DEPT COMPID
1 James Potter ICP 1001
2 Ethan McCarty ETA NULL
3 Emily Rayner ETA 1002
4 Jack Abraham ETA NULL
5 Ayaz Mohammad ICP 1003
Computer Table
COMPID MAKE MODEL MYEAR
1001 Dell Vostro 2013
1002 Dell Precision 2014
1003 Lenovo Edge 2013
1004 Lenovo Horizon 2014
1005 Apple MacPro 2015
SELECT Make,COUNT(*) FROM Employee E RIGHT OUTER JOIN Computer C ON E.CompId = C.CompId
WHERE Model <> 'Vostro' GROUP BY Make HAVING COUNT (ID) = 1 ORDER BY Make
Prev F J W G H S D O Next
Result
MAKE COUNT(*)
Lenovo 2
Dell 1
The SELECT clause is evaluated to remove all columns other than Make, COUNT(*).


Programs
What do you want to learn?
Outer Join
Font size
16
Page 5 of 5
Query Execution Order: Example 2
Employee Table
ID ENAME DEPT COMPID
1 James Potter ICP 1001
2 Ethan McCarty ETA NULL
3 Emily Rayner ETA 1002
4 Jack Abraham ETA NULL
5 Ayaz Mohammad ICP 1003
Computer Table
COMPID MAKE MODEL MYEAR
1001 Dell Vostro 2013
1002 Dell Precision 2014
1003 Lenovo Edge 2013
1004 Lenovo Horizon 2014
1005 Apple MacPro 2015
SELECT Make,COUNT(*) FROM Employee E RIGHT OUTER JOIN Computer C ON E.CompId = C.CompId
WHERE Model <> 'Vostro' GROUP BY Make HAVING COUNT (ID) = 1 ORDER BY Make
Prev F J W G H S D O Next
Result
MAKE COUNT(*)
Dell 1
Lenovo 2
Finally ORDER BY clause is evaluated to sort the result by Make in ascending order.

SQL JOIN ON THE SAME COLUMN

SQL JOIN

PRACTICE QUESTION

Subquery

What is a Subquery

In SQL, a subquery is a query within another query. It is a SELECT statement that is nested inside another SELECT, INSERT, UPDATE, or DELETE statement. The subquery is executed first, and its result is then used as a parameter or condition for the outer query.

Note - The topic is slightly difficult and needs a lot of practice Example - Find the movie with highest rating

Example - Find the movie with highest rating

name	rating	genre	year	released	score	votes	director	writer
The Shining	R	Drama	1980	June 13, 1980 (United States)	8.4	927000.0	Stanley Kubrick	Stephen King
The Blue Lagoon	R	Adventure	1980	July 2, 1980 (United States)	5.8	65000.0	Randal Kleiser	Henry De Vere Stacpoole
Star Wars: Episode V - The Empire Strikes Back	PG	Action	1980	June 20, 1980 (United States)	8.7	1200000.0	Irvin Kershner	Leigh Brackett
Airplane!	PG	Comedy	1980	July 2, 1980 (United States)	7.7	221000.0	Jim Abrahams	Jim Abrahams
Caddyshack	R	Comedy	1980	July 25, 1980 (United States)	7.3	108000.0	Harold Ramis	Brian Doyle-Murray
Friday the 13th	R	Horror	1980	May 9, 1980 (United States)	6.4	123000.0	Sean S. Cunningham	Victor Miller
The Blues Brothers	R	Action	1980	June 20, 1980 (United States)	7.9	188000.0	John Landis	Dan Aykroyd
Raging Bull	R	Biography	1980	December 19, 1980 (United States)	8.2	330000.0	Martin Scorsese	Jake LaMotta
Superman II	PG	Action	1980	June 19, 1981 (United States)	6.8	101000.0	Richard Lester	Jerry Siegel
The Long Riders	R	Biography	1980	May 16, 1980 (United States)	7.0	10000.0	Walter Hill	Bill Bryden
Any Which Way You Can	PG	Action	1980	December 17, 1980 (United States)	6.1	18000.0	Buddy Van Horn	Stanford Sherman

```
SELECT * FROM sql_cx_live.movies
WHERE score = (SELECT MAX(score) FROM sql_cx_live.movies)
```

What is the scope of inner query?

Types of Subqueries

Based on:

1. The result it returns

Scalar Subquery. (It is inner query which returns only one value (Either Text or Number))

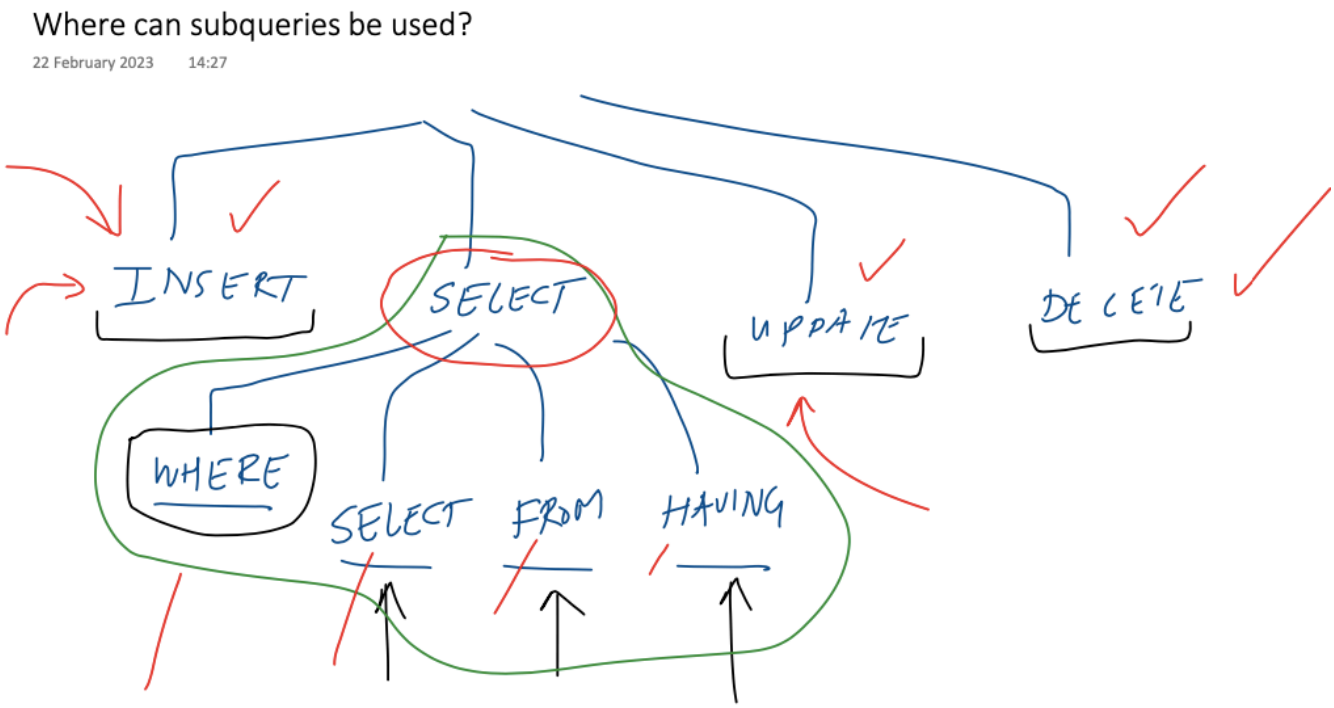
Row Subquery. (It is inner query which returns one columns means having multiple row in it.)

Table Subquery. (It is inner query which returns one table having more than 1 columns and more than one row.)

2. Based on working

Independent Subquery (It is that query in which its inner query is not depending on it's outer query.)

Correlated Subquery (It is that query in which its inner query is depending on its outer query.)



Independent Subquery - Scalar Subquery

1. Find the movie with highest profit(vs order by)

2. Find how many movies have a rating > the avg of all the movie ratings(Find the count of above average movies)

3. Find the highest rated movie of 2000

4. Find the highest rated movie among all movies whose number of votes are > the dataset avg votes

Independent Subquery - Row Subquery(One Col Multi Rows)

1. Find all users who never ordered
2. Find all the movies made by top 3 directors(in terms of total gross income) 3. Find all movies of all those actors whose filmography's avg rating > 8.5(take

25000 votes as cutoff)

Independent Subquery - Table Subquery(Multi Col Multi Row)

1. Find the most profitable movie of each year
2. Find the highest rated movie of each genre votes cutoff of 25000
3. Find the highest grossing movies of top 5 actor/director combo in terms of

total gross income

Correlated Subquery

1. Find all the movies that have a rating higher than the average rating of movies in the same genre. [Animation]

2. Find the favorite food of each customer.

Usage with SELECT

1. Get the percentage of votes for each movie compared to the total number of votes.

2. Display all movie names ,genre, score and avg(score) of genre -> Why this is inefficient?

Usage with FROM

1. Display average rating of all the restaurants

Usage with HAVING

1. Find genres having avg score > avg score of all the movies

Subquery In INSERT

Populate a already created loyal_customers table with records of only those customers who have ordered food more than 3 times.

Subquery in UPDATE

Populate the money col of loyal_cutomer table using the orders table. Provide a 10% app money to all customers based on their order value.

Subquery in DELETE

Delete all the customers record who have never ordered.

Transactions

What are Transactions?

A database transaction is a sequence of operations that are performed as a single logical unit of work in a database management system (DBMS). A transaction may consist of one or more database operations, such as inserts, updates, or deletes, which are treated as a single atomic operation by the DBMS. We can write transaction in Functions & Store Procedure which we will learn later on.

It follows the principle of all or none.

What is Commit, Rollback and Savepoint?

In a database transaction, there are three main commands that are used to manage the transaction:

1. **Commit:** A commit command is used to permanently save the changes made by a transaction to the database. When a transaction is committed, all changes made by the transaction are made permanent and cannot be rolled back.

2. **Rollback:** A rollback command is used to undo the changes made by a transaction and return the database to its state before the transaction began. When a transaction is rolled back, all changes made by the transaction are discarded and the database is returned to its previous state.

3. **Savepoint:** A savepoint command is used to mark a specific point within a transaction where a rollback can be performed. This allows for partial rollbacks of a transaction, where only changes made after the savepoint are undone, while changes made before the savepoint are still committed to the database.

What is Autocommit?

Autocommit is a feature of database management systems (DBMS) that automatically commits each individual database transaction as soon as it is completed, rather than requiring an explicit commit command to be issued.

When Autocommit is enabled, each individual SQL statement issued against the database is treated as a separate transaction and is committed immediately after it is executed. This means that each SQL statement becomes a separate, independent transaction, and its effects are immediately visible to other users.

Each SQL write statement is Autocommit (prove)

set Autocommit = 0 and (show)

START TRANSACTION -> show operations without committing

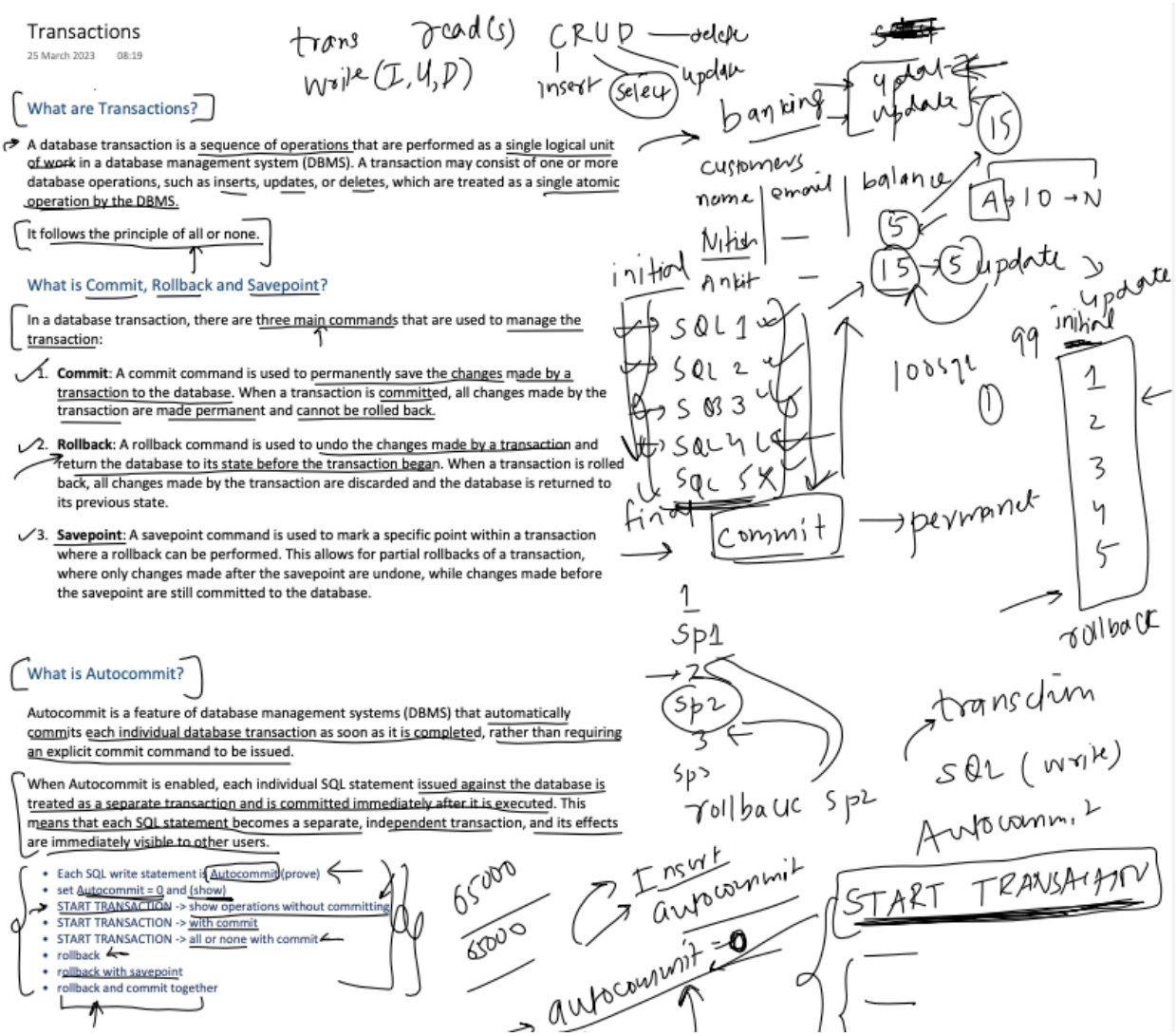
START TRANSACTION -> with commit

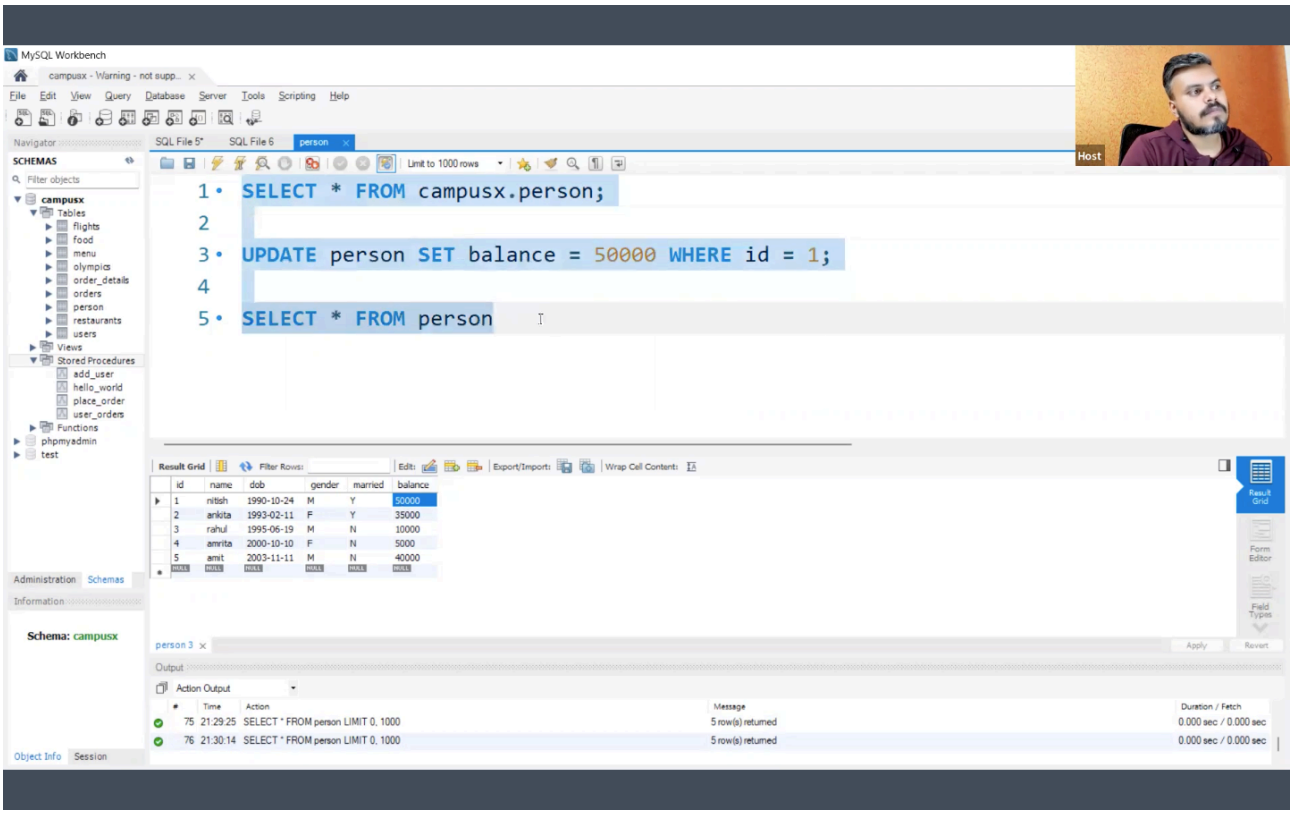
START TRANSACTION -> all or none with commit

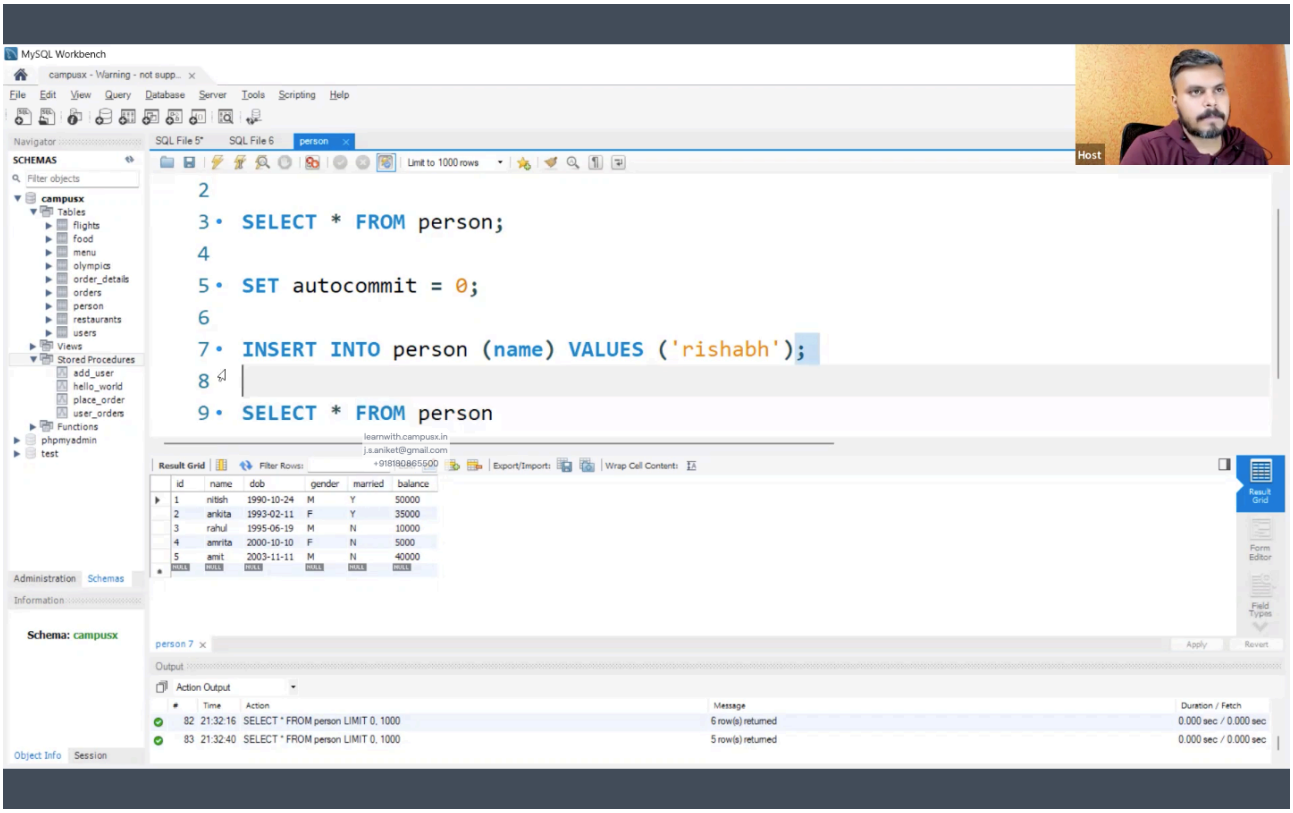
rollback

rollback with savepoint

rollback and commit together

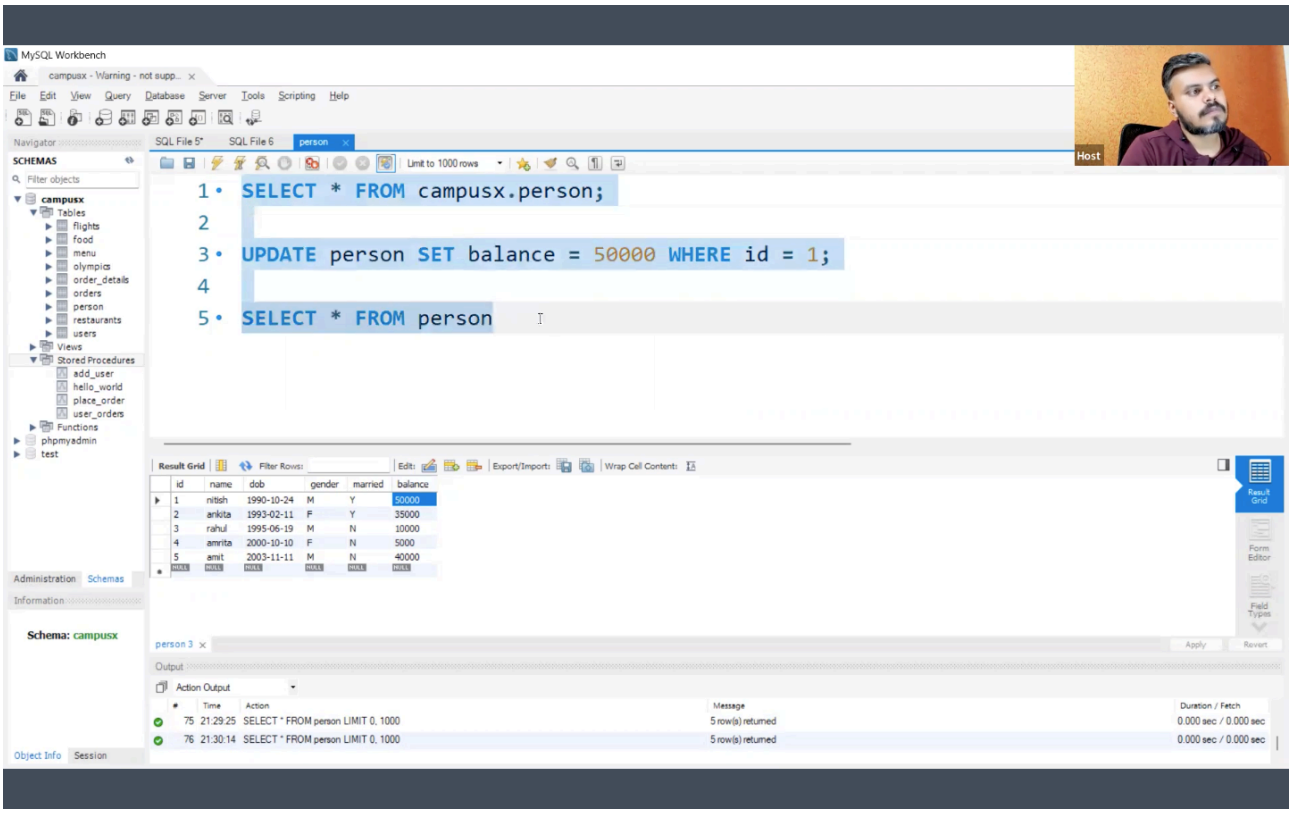


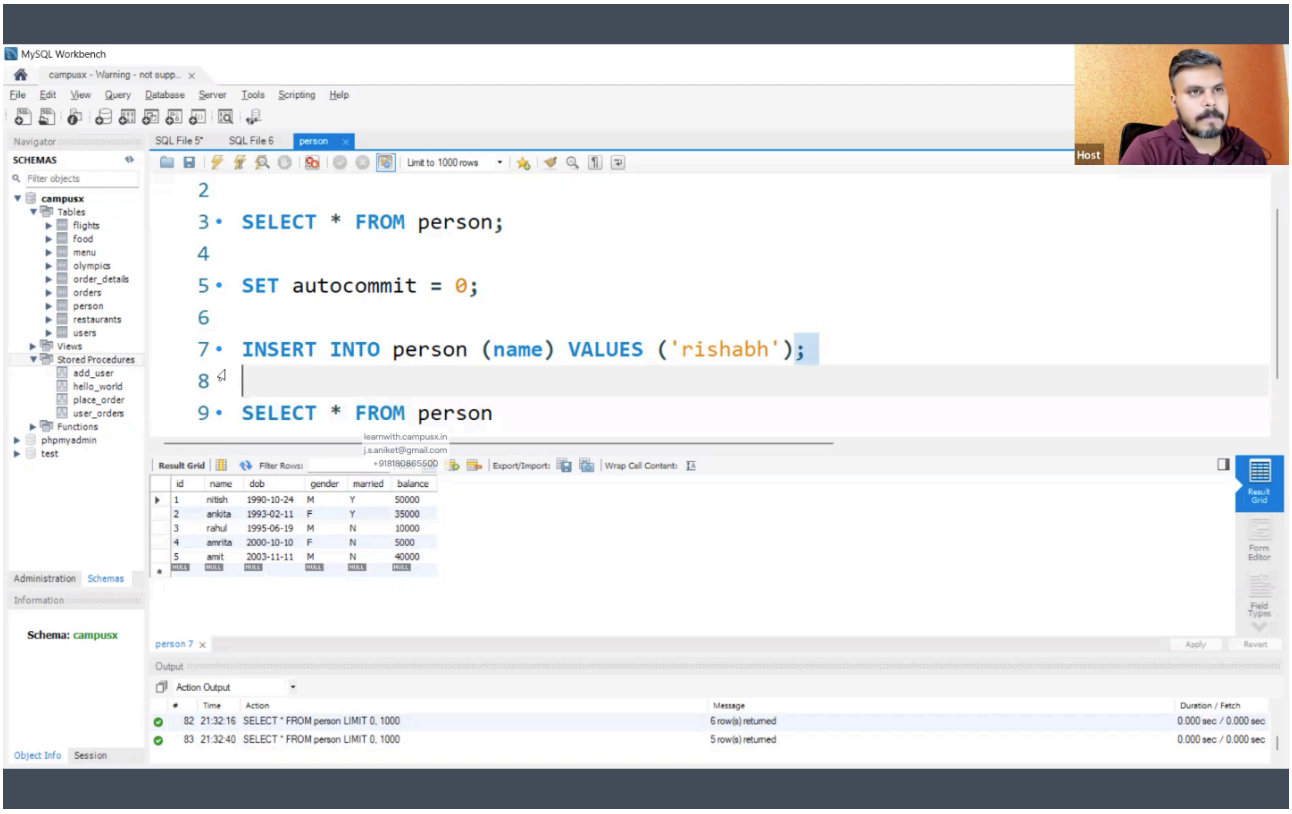


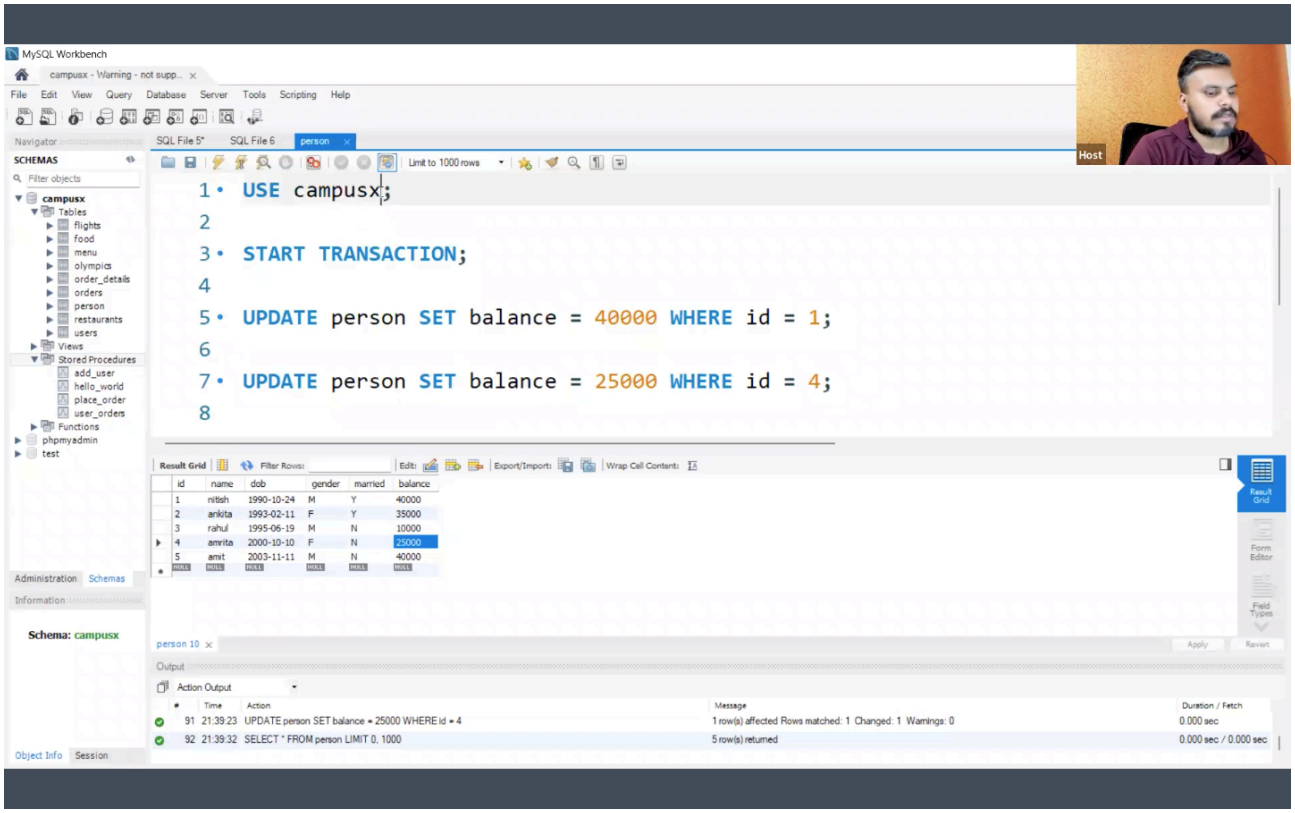


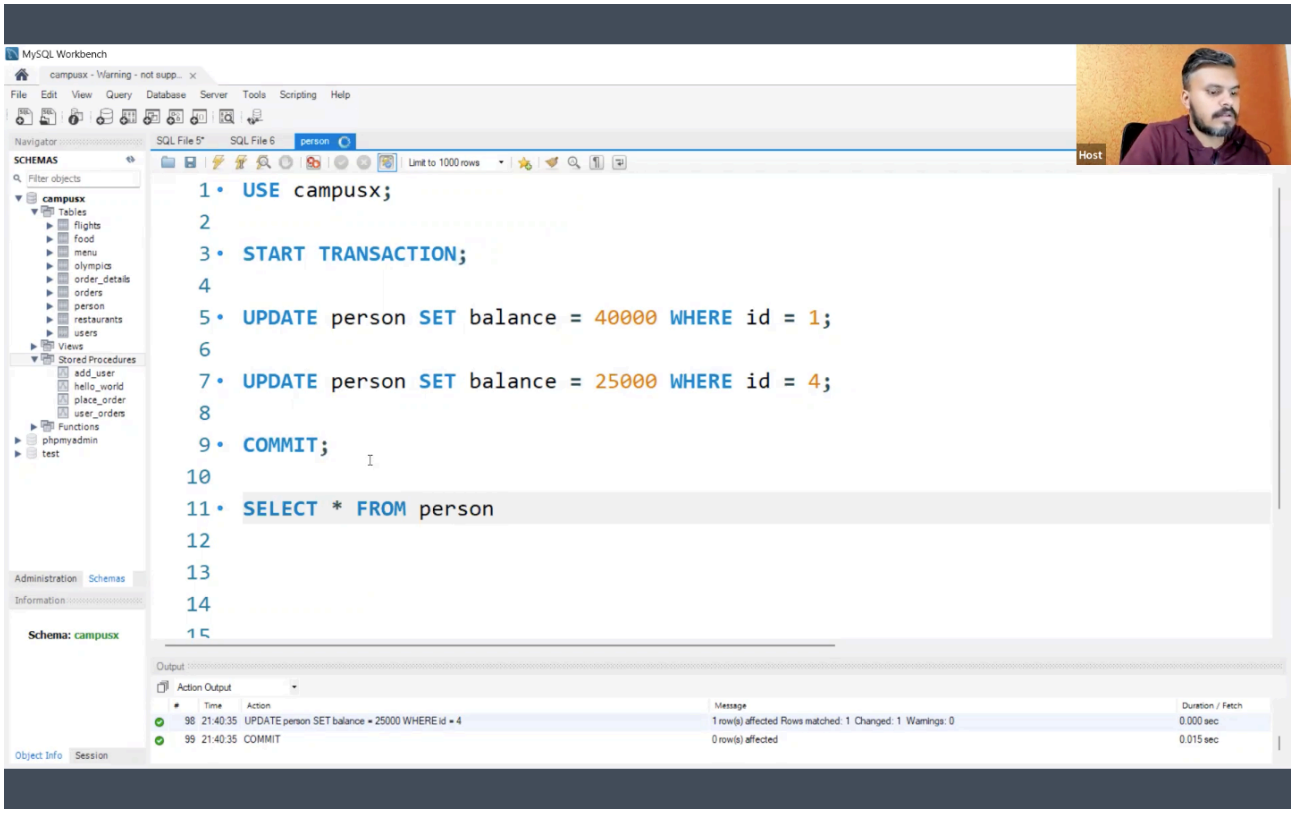
Autocommit=0 means we are putting off the Autocommit function.

Autocommit=1 means we are putting on the Autocommit function.


MySQL Workbench
SELECT * FROM campusx.person;
UPDATE person SET balance = 50000 WHERE id = 1;
SELECT * FROM person
Host


MySQL Workbench
SELECT * FROM person;
SET autocommit = 0;
INSERT INTO person (name) VALUES ('rishabh');
SELECT * FROM person
Host


MySQL Workbench
USE campusx;
START TRANSACTION;
UPDATE person SET balance = 40000 WHERE id = 1;
UPDATE person SET balance = 25000 WHERE id = 4;
Host


MySQL Workbench
USE campusx;
START TRANSACTION;
UPDATE person SET balance = 40000 WHERE id = 1;
UPDATE person SET balance = 25000 WHERE id = 4;
COMMIT;
SELECT * FROM person
Host

What is ACID properties of a Transaction?

ACID is an acronym that stands for Atomicity, Consistency, Isolation, and Durability, which are a set of properties that ensure reliable database transactions:

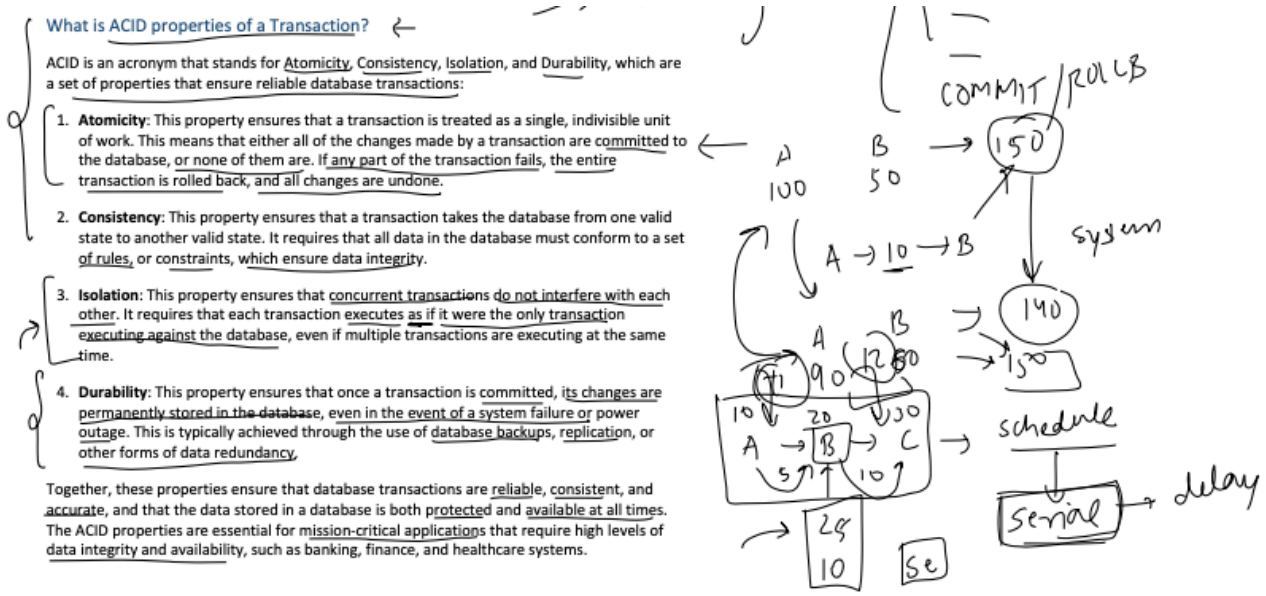
1. **Atomicity:**This property ensures that a transaction is treated as a single, indivisible unit of work. This means that either all of the changes made by a transaction are committed to the database, or none of them are. If any part of the transaction fails, the entire transaction is rolled back, and all changes are undone.

2. **Consistency:** This property ensures that a transaction takes the database from one valid state to another valid state. It requires that all data in the database must conform to a set of rules, or constraints, which ensure data integrity.

3. **Isolation:** This property ensures that concurrent transactions do not interfere with each other. It requires that each transaction executes as if it were the only transaction executing against the database, even if multiple transactions are executing at the same time.

4. **Durability:** This property ensures that once a transaction is committed, its changes are permanently stored in the database, even in the event of a system failure or power outage. This is typically achieved through the use of database backups, replication, or other forms of data redundancy.

Together, these properties ensure that database transactions are reliable, consistent, and accurate, and that the data stored in a database is both protected and available at all times. The ACID properties are essential for mission-critical applications that require high levels of data integrity and availability, such as banking, finance, and healthcare systems.



WINDOW FUNCTIONS

PL-SQL

STORE PROCEDURE

Functions

User Defined Functions

- What are function and their advantages

User-defined functions (UDFs) in SQL are functions that are created by users to perform specific tasks. These functions can be used just like built- in functions in SQL and can take parameters as input, perform some operations on them, and then return a value.

- Syntax

```
DELIMITER $$

CREATE FUNCTION Function_Name(
    Parameter_1 DataType,
    Parameter_2 DataType,
    Parameter_n DataType,
)
RETURNS Return_Datatype
[NOT] DETERMINISTIC
BEGIN
        Function Body
        Return Return_Value
END $$

DELIMITER ;
```

Examples

- hello world
- parameterised vs non parameterised
- Calculate age in years(for col)
- greet with name -> conditional title
- Date formatting and flights between 2 cities(deterministic Vs Non Deterministic)
- show all functions of a database
- Drop function

Benefits

Simplifies SQL queries

Reusability

Enhances readability

VIEWS