Query 1: Insert a New Winner into the Winners Table Why this query is used: This query adds a new winner (Ada Lovelace) into the Winners table, including relevant details such as name, gender, birth date, country, and affiliation.

```
INSERT INTO Winners (winner_id, first_name, last_name, gender, date_of_birth, date_of_death, country_id, affiliation_id)

VALUES (11, 'Ada', 'Lovelace', 'Female', '1815-12-10', NULL, 1, 5);
```

Query 2: Insert a New Prize Record Why this query is used: This query inserts a new prize entry awarded in 2023 to Ada Lovelace for her contributions to computing.

```
INSERT INTO Prizes (prize_id, year_awarded, category_id, winner_id, citation)

VALUES (11, 2023, 1, 11, 'For pioneering contributions to early computing and programming.');
```

Query 3: Update Winner's Affiliation Why this query is used: This query modifies Ada Lovelace's affiliation ID in the Winners table to reflect a new affiliation.

```
UPDATE Winners SET affiliation_id = 3 WHERE winner_id = 11;
```

Query 4: Update Media Coverage Title Why this query is used: This query updates the title of a media coverage entry to reflect Bob Dylan's Nobel Laureate status.

```
UPDATE MediaCoverage SET title = 'Bob Dylan: A Nobel Laureate in Music' WHERE media_id = 9;
```

Query 5: Delete a Specific Media Coverage Entry Why this query is used: This query removes a media coverage entry with media_id 10 from the MediaCoverage table.

```
DELETE FROM MediaCoverage WHERE media_id = 10;
```

Query 6: Retrieve Female Winners Why this query is used: This query selects only female winners from the Winners table.

```
SELECT first_name, last_name, gender, country_id FROM Winners WHERE gender = 'Female';
```

Query 7: Retrieve Winners and Their Prizes Why this query is used: This query joins the Winners and Prizes tables to retrieve winners' names and the prizes they have won.

```
SELECT w.first_name, w.last_name, p.year_awarded, c.category_name

FROM Winners w

JOIN Prizes p ON w.winner_id = p.winner_id

JOIN Categories c ON p.category_id = c.category_id;
```

Query 8: Retrieve Media Coverage for Prizes Why this query is used: This query retrieves media coverage details, including the title, source, year awarded, and category name.

```
SELECT mc.title, mc.source, p.year_awarded, c.category_name
```

FROM MediaCoverage mc

JOIN Prizes p ON mc.prize_id = p.prize_id

JOIN Categories c ON p.category_id = c.category_id;

Query 9: Count Total Winners by Country Why this query is used: This query counts the total number of winners from each country.

SELECT country_id, COUNT(*) AS total_winners FROM Winners GROUP BY country_id;

Query 10: Count and Average Year of Prizes by Category Why this query is used: This query calculates the total number of prizes awarded and the average year of those prizes by category.

SELECT c.category_name, COUNT(p.prize_id) AS total_prizes, AVG(p.year_awarded) AS avg_year

FROM Prizes p

JOIN Categories c ON p.category_id = c.category_id

GROUP BY c.category_name;

Query 11: Retrieve Detailed Winner and Prize Information Why this query is used: This query retrieves winners' details, their country, prize information, and related media coverage.

SELECT w.first_name, w.last_name, c.country_name, p.year_awarded, cat.category_name, mc.title AS media_title

FROM Winners w

JOIN Countries c ON w.country_id = c.country_id

JOIN Prizes p ON w.winner_id = p.winner_id

JOIN Categories cat ON p.category_id = cat.category_id

LEFT JOIN MediaCoverage mc ON p.prize_id = mc.prize_id;

Query 12: List Winners Affiliated with Harvard University Why this query is used: This query retrieves winners who were affiliated with Harvard University.

SELECT w.first_name, w.last_name, a.institution_name

FROM Winners w

JOIN Affiliations a ON w.affiliation_id = a.affiliation_id

WHERE a.institution_name = 'Harvard University';

Query 13: Retrieve Winners from a Specific Country Why this query is used: This query selects all winners from a specific country (Country ID = 1).

SELECT * FROM Winners WHERE country_id = 1;

Query 14: Create an Index on Winners Table Why this query is used: This query creates an index on the country_id column to optimize searches.

CREATE INDEX idx_winner_country ON Winners(country_id);

Query 15: Explain Query Using Index Why this query is used: This query checks how an indexed query execution plan differs from a non-indexed query.

EXPLAIN SELECT * FROM Winners WHERE country_id = 1;

Query 16: Create a View for Winner Details Why this query is used: This query creates a view that combines winner names, country names, and affiliations for easier access.

CREATE VIEW WinnerDetailsView AS

SELECT w.winner_id, CONCAT(w.first_name, ' ', w.last_name) AS full_name, c.country_name, a.institution_name

FROM Winners w

JOIN Countries c ON w.country_id = c.country_id

LEFT JOIN Affiliations a ON w.affiliation_id = a.affiliation_id;

Query 17: Grant User Privileges Why this query is used: This query grants different levels of privileges to users for database security and role management.

GRANT ALL PRIVILEGES ON nobelprizedata.* TO 'anusha'@'localhost' IDENTIFIED BY 'AnushaPass';

GRANT SELECT ON nobelprizedata.* TO 'nikhi'@'localhost' IDENTIFIED BY 'NikhiPass';

GRANT INSERT ON nobelprizedata.Winners TO 'maheshwari'@'localhost' IDENTIFIED BY 'MaheshwariPass';

Qyery 18 : Stored Procedure to Retrieve Nobel Prize Winner Details by Category

The following stored procedure fetches details of Nobel Prize winners based on a given category:

CREATE PROCEDURE GetWinnerDetails(IN category_name VARCHAR(64))

BEGIN

SELECT

w.winner_id,

w.first_name,

w.last_name,

c.category_name,

```
    p.year_awarded
  FROM
    Winners w
  JOIN
    Prizes p ON w.winner_id = p.winner_id
  JOIN
    Categories c ON p.category_id = c.category_id
  WHERE
    c.category_name = category_name;
END
```

Query 19: Function to Calculate Total Number of Nobel Prizes in a Given Category

The following SQL function calculates and returns the total number of Nobel Prizes awarded for a specified category:

```
CREATE FUNCTION GetTotalPrizes(category_name VARCHAR(64))
RETURNS INT
DETERMINISTIC
BEGIN
  DECLARE total_prizes INT;

  SELECT COUNT(*) INTO total_prizes
  FROM Prizes p
  JOIN Categories c ON p.category_id = c.category_id
  WHERE c.category_name = category_name;

  RETURN total_prizes;
END
```

Query 20: Trigger to Prevent Future Birth Dates for Winners

The following SQL trigger ensures that no winner's birth date is set in the future:

```
CREATE TRIGGER BeforeInsertWinner

BEFORE INSERT ON Winners

FOR EACH ROW

BEGIN

   IF NEW.date_of_birth > CURDATE() THEN

      SIGNAL SQLSTATE '45000'

      SET MESSAGE_TEXT = 'Error: Birth date cannot be in the future';

   END IF;

END
```

Query 21: Trigger to Update Total Prizes Count After Prize Insertion

The following SQL trigger updates the total number of prizes for a winner whenever a new prize is inserted into the Prizes table:

```
CREATE TRIGGER AfterInsertPrize

AFTER INSERT ON Prizes

FOR EACH ROW

BEGIN

   UPDATE Winners

   SET total_prizes = total_prizes + 1

   WHERE winner_id = NEW.winner_id;

END
```