**The key SQL set operations are:**

 **UNION**

 **UNION ALL**

 **INTERSECT** (Not supported in all databases like MySQL)

 **EXCEPT** or **MINUS** (MySQL uses EXCEPT starting from version 8.0.31)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| .-- Students\_A Table   |  |  | | --- | --- | | student\_id | name | | 1 | Alice | | 2 | Bob | | 3 | Charlie |   . | . -- Students\_B Table   |  |  | | --- | --- | | student\_id | name | | 2 | Bob | | 3 | Charlie | | 4 | David |   . |

**1. UNION**

The UNION operator combines the results of two SELECT queries and removes duplicate rows. It essentially performs a **set union**.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| -- Query combining two tables using UNION  **SELECT** student\_id, name **FROM** Students\_A  **UNION**  **SELECT** student\_id, name **FROM** Students\_B; | .   |  |  | | --- | --- | | Student\_id | Name | | 1 | Alice | | 2 | Bob | | 3 | Charlie | | 4 | David |   . |

### 2. UNION ALL

UNION ALL works like UNION, but it **does not remove duplicates**. It returns all rows, including duplicates.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| -- Query combining two tables using UNION ALL  **SELECT** student\_id, name **FROM** Students\_A  **UNION ALL**  **SELECT** student\_id, name **FROM** Students\_B; | .   |  |  | | --- | --- | | Student\_id | Name | | 1 | Alice | | 2 | Bob | | 3 | Harlie | | 2 | Bob | | 3 | Charlie | | 4 | david |   . |

### 3. INTERSECT

INTERSECT returns only the rows that are **common** between the two SELECT queries, performing a **set intersection**. Unfortunately, MySQL does not natively support INTERSECT, but you can simulate it using JOIN.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| -- Simulating INTERSECT in MySQL using INNER JOIN  **SELECT** a.student\_id, a.name  **FROM** Students\_A a  **JOIN** Students\_B b **ON** a.student\_id = b.student\_id; | .   |  |  | | --- | --- | | Student\_id | name | | 2 | Bob | | 3 | Charlie |   . |

### 4. EXCEPT or MINUS

The EXCEPT (or MINUS in Oracle) operator returns the rows from the first SELECT query that are **not present** in the second SELECT query, performing a **set difference**.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| -- Query using EXCEPT (MySQL 8.0.31+)  **SELECT** student\_id, name **FROM** Students\_A  **UNION**  **SELECT** student\_id, name **FROM** Students\_B; | .  .   |  |  | | --- | --- | | Student\_id | Name | | 1 | Alice |   . |
| If you're using an older version of MySQL, you can simulate the EXCEPT operation with a LEFT JOIN:  -- Simulating **EXCEPT** using **LEFT JOIN** in MySQL  **SELECT** a.student\_id, a.name  **FROM** Students\_A a  **LEFT JOIN** Students\_B b **ON** a.student\_id = b.student\_id  **WHERE** b.student\_id **IS NULL;** |